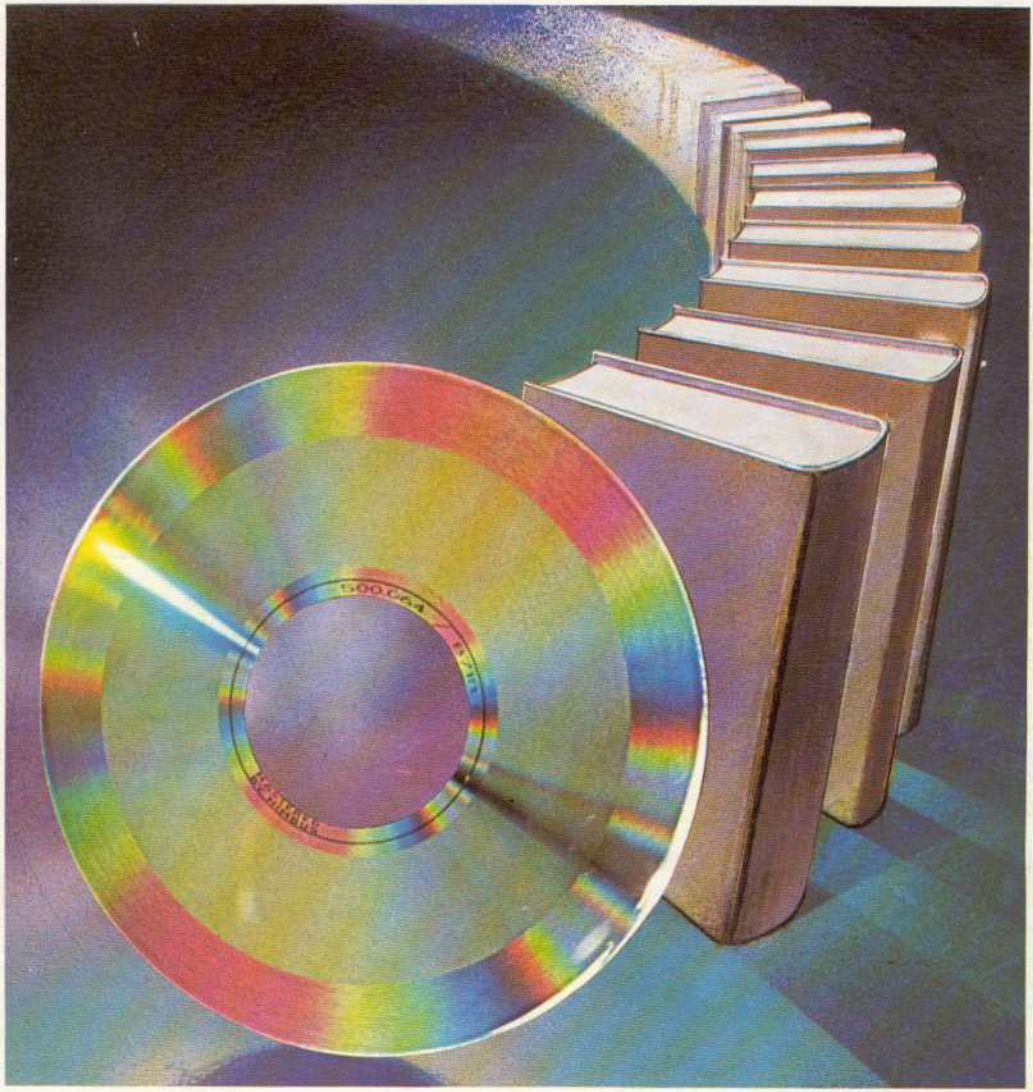
Ára: 30 Ft

mikro számítógép magazin



A JÖVŐ ÍGÉRETE

1988/1

Európa térkép

BPES üzemmód

A MUPID cikksorozat folytatásához a gazdag képi megjelenítés lehetőségeit mutatjuk be.
(Cikkünk a 22. oldalon.)

μ mikro számítógép magazin

6. ÉVFOLYAM
1988/1. SZÁM

A NEUMANN JÁNOS SZÁMÍTÓGÉPTUDOMÁNYI TÁRSASÁG ÉS A KISZ KÖZPONTI BIZOTTSÁG LAPJA

A kiadvány a Tudományszervezési és Informatikai Intézettel együttműködve készül

A szerkesztőbizottság vezetője:
Kovács Győző

A szerkesztőség munkatársai:
Babos János (tervezőszerkesztő)
Bakos Tamás (programozástechnika)
Broczkó Péter (hírek)
Énekes Ferenc (KISZ, Diákszerkesztőség)
Jakab Ágnes (olvasószerkesztő)
Kovács Győző (levelezés)
Lindner László (sakk)
Krasznai Éva
Petróczy Judit (könyvek)
Pinke György (NJSZT, alkalmazások)
Simonyi Endre
Szebenszki Sándor
Szulyovszky Csaba
Tamásné Lakó Erika
Terebessy Ákosné
Varga András (TII, iskola-számítógép)
Vizessy Mária

Címképünk:
Ramocsai Imri munkája

Felelős szerkesztő:
Könyves Tóth Pál

Szerkesztőség:
1027 Budapest, Fő u. 68.
Telefon: 154-250

Levélcím:
1371 Budapest
Pf. 433.

Kiadja az Ifjúsági Lap- és Könyvkiadó Vállalat

Felelős kiadó:
dr. Király G. István
igazgató

Kiadóhivatal:
1065 Budapest, Révay u. 16.
Telefon: 116-660

Terjeszti a Magyar Posta
Előfizethető a hírlapkézbesítő hivataloknál
és a Posta Hírlapelőfizetési és Lapeliátási Irodáján
(1900 Budapest V., József nádor tér 1.)
vagy átutalással a 215—96 162 pénzforgalmi jelzőszámra.

Megjelenik havonta
Egy szám ára 30,— Ft
Előfizetési díj:
egy évre 360,— Ft
fél évre 180,— Ft
Külföldön terjeszti a Kultúra,
1389 Budapest, Pf. 149.
és a Magyar Média
1932 Budapest, Pf. 279.
86—0253

Szikra Lapnyomda
Budapest (88—1824)
Felelős vezető:
Csöndes Zoltán vezérigazgató

INDEX: 25 629
ISSN 0236-6088

Tartalom

Belső tartalékaink 2
Rendszerfejlesztési eszközök. Szoftver 18
Szemétgyűjtés 19
Miért nem stupid a MUPID? 22
Optikai tárolás ma és holnap 24
Pályázati felhívások 26
1987. évi tartalomjegyzék 27
Szerviz a város peremén 30
Olvastunk . . . 38
μINFORM 39
Újföld 40
Adok — veszek — cserélek 43

ISKOLA-SZÁMÍTÓGÉP

TechnoMIR 3
Graftálnövények 4

DIÁKROVAT

A LIST parancs titkosítása C64-re 6
A beszélő Spectrum 6
Bővített BASIC HT-re 7
Rövid játék VIC—20-ra 7
Kapcsoltam? Kapcsoltam! 8
Szerkesztőtársunk mondd, akarsz-e lenni? 9

PROGRAMOZÁSTECHNIKA

BASIC és gépi kód 10
Z80 programok haladóknak Spectrumra és Primóra 11
Számítógépes grafika Pascalban. Kifestés 13
Függvények és utasítások II. 16

μKLUB

Klubsirató 33
C64 modem 34

SAKK

A játékfa és kiértékelése 42

JÁTÉK

Fordítós játék 44

AZ OLVASÓ ÍRJA 46

KÖNYVEK 47

HÍREK, ÉRDEKESSÉGEK 48

„Belső tartalékaink között kiemelt hely illeti meg a tudást, a munkaerőkölcsön és -kultúrát, az irányítás és a vezetés színvonalát. Ebben az összefüggésben is jelentős szerepet tulajdonítunk a tudománynak, a munkaerő magas szintű képzésének, a tudás nagyobb társadalmi és anyagi megbecsülésének. Számunkra egyértelmű, hogy a szűkös beruházási feltételek közepette az oktatás az egyik leggazdaságosabb termelő célú befektetés. A hatékonyság növelése azonban e területen is elengedhetetlen."
(Grósz Károly miniszterelnök parlamenti expozéja. 1987. szept. 16.)

Belső tartalékaink...

Lassan ötödik éve jelentkezem már hónapról hónapra egy-egy szerkesztőségi irással a µM negyedik oldalán, mindig is arra törekedve, hogy egy éppen aktuális problémáról mondjam el a véleményemet.

Új év kezdetén a szokásosnál is nehezebb a dolgom, hiszen ilyenkor programot kell adni, irányt mutató elhatározásokról illik szólni, el kell mondani, hogy mint szeretnénk újabb ötletekkel segíteni a mindmáig is elsőrendű célunkat, a társadalom informatizálását.

A kedves olvasó már a politikai töltésű, miniszterelnöki idézetből is sejheti, hogy most nem a Magazinról szeretnék eszmét futtatni; sokkal inkább a magyar informatika problémáiról és az expozé — véleményem szerint — legfontosabb megállapításáról, nevezetesen arról, hogy ez az ország a következő években csak akkor boldogulhat, ha a *meglévő tartalékokat* nemcsak felfedjük, de — *amilyen gyorsan csak lehet — fel is szaghatjuk*.

Az informatika, a számítástechnika, az elektronizálás sincs kivételes helyzetben, és — úgy gondolom — egyáltalán nem különbözik a népgazdaság más területeitől: ha nem is bőséges, de azért jelentős tartalékokkal rendelkezik.

A miniszterelnöki program szerint az új évet indító rendelkezésekkel — beleértve még az új adózási rendszert is — a kormány nemcsak a *vállalati*, de az *állampolgári magatartást* is *befolyásolni* szándékozik. Az én értelmezésem szerint ez azt is jelenti, hogy talán felszámolható lesz a közös ügyeink iránti, ma még meglévő nagyfokú közömbösség.

Egyszer egy jogász barátom azt mondta (lehet, hogy ez egyáltalán nincs így), hogy hazai jogrendszerünk nem ismeri el az elmulasztott lehetőségekből keletkezett károkat, és így azt meg sem lehet téríttetni. Egy példával világította meg a dolgot. Képzeljük el, hogy szervizbe adtuk a kocsit, megkapjuk, és elmegyünk vele egy fontos vidéki megbeszélésre. A kocsi a szerviz hibájából valahol a pusztán leáll, a megbeszélést lekéssük, emiatt az általunk képviselt cég elesik kétmillió forint árbevételtől. A szerviz ezután ingyen vállalja a kocsi ismételt javítását, de az elmaradt haszon, a kétmillió forint kifizetését visszautasítja. Ha bírósághoz fordulok, egész biztos, hogy nem fogja megítélni a nyilvánvaló kárt.

Nos, én azt hiszem, hogy a magyar informatika tartalékai éppen az évek hosszú során összegyűjtött „elmaradt hasznok" sorozatos felszámolásában vannak. Azt ma már az elemista gyerek is tudja, hogy a termelés, az adminisztráció, az ügykezelés, a pénzforgalom, de az *oktatás* korszerűsítése, hatékonyságának növelése is *csak* korszerű, intelligens elektronikus eszközökkel képzelhető el. Nagyon jól érzékelhető az is, de valójában nem tudom megítélni, hogy a kormány meghirdetett reformtörekvéseinek mikor lesz szinte áttörhetetlen gátja nemcsak a sokat emlegetett telefonvonalak, de az országos számítógéphálózat hiánya, ami miatt például már ma sem tudunk bekapcsolódni az üzleti élet kívánta nemzetközi kommunikációs rendszerbe. Vagy mikor okoz konfliktust az „elektronikus pénz" lassú bevezetése, például a hitelkártyarendszer létrehozásával kapcsolatos tétovázás, ami miatt a pénzkezelésben nem lehet a különben felesleges munkaerőt felszabaditani, és ami miatt fél napokat tölt el ki tudja mennyi „aktív munkaerő" a pénzintézetek előtti sorban állással.

Még mindig a pénznél maradva, már régen szakítani lehetett volna a fizetések és egyéb jövedelmek pénztáraknál, boritékban való felvételével, ami pénzszállítással, boritékolással, nyugták, átvételi elismervények és ezzel együtt a haszontalan papírhegyek növelésével jár. Szerintem erre azonnal óriási igény lenne, ha képes volna a pénzünket kezelő bank tisztességes és részletes átutalási jelentés készítésére, amelyre ma már nemcsak kíváncsiságból, hanem az adózási kötelezettség tisztességes teljesítése miatt is föltétlenül szükség van.

A felsorolt példákkal azt szerettem volna bemutatni, hogy a szánaitástechnika alkalmazásában is vannak még olyan „szűz" területek, amelyeken a gépesítés nemcsak gazdaságos, de a társadalmi és politikai haszna is beláthatatlan.

Egyszer egy itthoni konferencián, amelyen számos külföldi és hazai előadás is elhangzott, egy angol barátom megjegyezte: „Ti, magyarok, nagyon szeniális nép vagytok, de mi szükség van arra, hogy olyat is felfedezzetek és kutassatok, amit már mások régen alkalmaznak?". Egy korábbi szerkesztőségi cikkemben az NJSZT országos kongresszusa alkalmából eszmét futtattam erről a szerintem speciális hazai problémáról (µM 1987/2. szám). Szintúgy a szoftverlopásról is (1987. 10. szám), aminek, bármennyire is furcsa, a gyökerét ugyanott lehet keresni, mint a megismételt fejlesztéseknek. Mind az újrafejlesztésre, mind a lopásra megvan a tetszetős magyarázat. Azt mondják ugyanis, hogy lehetőleg ne vegyük meg a készet, mert devizába kerül, nem alkalmazható a speciális magyar viszonyokra, meg aztán a készet átvenni nem nagy dicsőség, mert az szolgai alkalmazás; sokkal nagyobb dolog valamit fejleszteni, és akkor még nem beszéltünk a nagyobb anyagi elismerésről, ti. nálunk a munkát és nem a teljesítményt fizetik. Ha meg valami ellopható, akkor törjük fel a védelmet, utána meg toldozgassuk-foldozgassuk a rendszert egészen addig, amíg az eredeti fejlesztő, a tulajdonos már külföldi bemutatókon sem ismer a saját „gyerekére". Igaz, hogy így gyengülnek nemzetközi kapcsolataink, rossz lesz a piaci megítélésünk, elveszítjük szakmai hitelünket, de azért fájjon a hatóságok feje; az ilyen felfogású embereket ugyanis ezek a mellékörülmények nem érdeklik.

Pedig ennek a felfogásnak a megváltoztatása is egyike belső tartalékainknak, hiszen nem egy intézet, vállalat, gmk, pjt eredményei azt mutatják, hogy nemcsak lopott, hanem eredeti gondolatok megvalósításával is tudunk eredményt elérni. Sőt, az eredeti gondolatok értéke az iparilag fejlett országok piacán sokkal magasabb, és még itthon is nagyobb haszonnal jár, mint amit illegális tevékenységgel bármikor elérhetünk, nem is beszélve a nem forintosítható vagy nem dollárosítható előnyökről.

Nyilvánvaló, hogy egy szerkesztőségi cikk keretében, ami már így is túl hosszúra sikeredett, csak érintőlegesen lehet a témát érinteni, minden részletről beszélni reménytelen vállalkozás. Ezt a témakört azért anélkül mégsem lehet lezárni, hogy néhány szót ne szóljak az érem másik oldaláról is.

Sok esetben ugyanis az ismert rendszereket itthon azért fejlesztik ki újból, vagy ne adj' isten, azért lopják el, mert legális beszerzésükre, elsősorban devizás gondjaink miatt, egyáltalán nincs lehetőség. A kormány programja — ha jól olvastam — úgy látszik, mintha a devizagazdálkodásban is új utakat akarna nyitni, amire valóban igen nagy szükség van. A szakma tele van történetekkel az elhúzódó importengedélyekről, kompenzációs üzletek leállításáról, egyáltalán a külpiaci kívánalmakkal kapcsolatos hosszú reakcióidőkről, ami a gazdaság minden területén, de a számítástechnikában különösen súlyosan akadályozza a szakma fejlődését. Azt hiszem, ennek felszámolása sem lebecsülendő „belső tartalékunk".

Mondanivalóm végére hagytam az indító miniszterelnöki idézetben említett *tudást, szakismeretet, munkaerőkölcsönt* és *munkakultúrát*, az *irányítás* és *vezetés színvonalát*. Ha őszinték akarunk lenni — és miért ne lennénk azok —, az elmúlt tíz évben a tudás nálunk alaposan devalválódott. A tanulás, különösen a folyamatos tanulás, egyéni hobbivá züllött, a tanult ember megbecsülése a lehető legmélyebbre süllyedt. Tudom, súlyosak a szavak, de ez az igazság. Szinte észrevétlenül kialakult egy olyan társadalom, amelyben a tanulást és főleg a tudást nem az előbbrejutás igénye motiválta; sokkal inkább a divat vagy családi tradiciók kötelezték a gyereket, hogy ő is, mint az apuka vagy a nagypapa, orvos, mérnök vagy tanár legyen. Aztán, amikor megvolt a diploma, és lakást, autót, nyaralót kellett vásárolni, két-három gyerekről kellett gondoskodni, tehát a tisztes élethez szükséges körülményeket kellett megteremteni, akkor a friss diplomás délu án felvette a kalapácsot, a malteres vödröt, szobafestést vállalt vagy villanyt szerelt. Ha a felsorolt javakat gyorsabban akarta megteremteni, akkor butikot nyitott vagy lángossütőt, és ha nagyon szerencsés volt, akkor elment benzinkutasnak.

Úgy vélem, hogy a szeilemi munkásokban rejlő belső tartalékok felszabaditására a kormány munkaprogramjától várható intézkedések megfelelő garanciát biztositanak. Ahhoz azonban, hogy az elmúlt években elmaradt lehetőségeket felszámolhassuk, ahhoz „... elsősorban tartalmi szempontból fontos áttekinteni oktatási rendszerünket. (...) Minden területen honorálni, ösztönözni kell a szakmai és általános ismeretek folyamatos gyarapítását. Ehhez a felnőttképzés, átképzés és továbbképzés jelenleginél hatásosabb rendszerének kiépitése szükséges." (Idézet az expozéból.)

Erre szokták azt mondani, hogy *így igaz*. Tételezzük fel ugyanis, hogy a kormány rövid időn belül a kívánt mértékben megteremti a szellemi munka társadalmi, anyagi megbecsülését, és a felnőtt társadalom néhány hónap alatt motiválva érzi magát, és a programnak megfelelően egyszerre mindenki továbbképzésre fog jelentkezni. Szerintem rövidesen kiderülne, hogy a társadalmi méretű továbbképzéshez a megfelelő intézményrendszerrel, képzési technológiával, a követelményeknek megfelelő, gyorsan és rugalmasan változó oktatási anyagokkal egyáltalán nem rendelkezünk.

A mai merev, méltán porosznak „becézett" oktatási rendszerben a diák igen kevés számú, igen sok kötelező előírással terhelt pálya között „válogathat", az oktatási intézmény pedig keretszámokkal korlátozottan, nagyon nagy hibaszázalékú felvételi vizsgákkal próbálkozik a megfelelő és az adott pályára alkalmas, tehetséges diákok kiválasztásával. A küzdelem szisizfuszinak látszik, az oktatási rendszer reformja nélkül a gazdasági reform, amelyhez folyamatosan tovább képzett, jó szakemberek kellenek, nem vagy csak nagyon nehezen lesz megvalósítható. Az oktatási reform pedig az én értelmezésemben nyílt és technologizált oktatási rendszert, távtanulást, tömeges és folyamatos képzést jelent.

Ebben az újévi irásomban egyet szerettem volna annak a nem kevés pesszimista honfitársamnak elmondani, akik egyelőre egykedvűen szemlélik az ország felemelkedésére tett törekvéseket: tenni valamit mindannyiunkért nemcsak a kormány dolga. Ha valamennyien körülnézünk a házunk táján, és megpróbálva a lehetetlent is, kihasználjuk a belső tartalékokat, akkor mindannyiunknak könnyebb lesz az élete.

Ebben a reményben kíván boldog új esztendőt

KOVÁCS GYŐZŐ

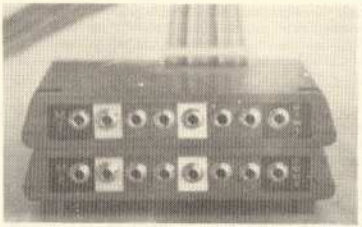
TechnoMIR

Digitális kimenet

A DOUT (digitális kimenet) modul a DIN (digitális bemenet) egységgel egyenrangú: ha a DIN a számítógép szeme, a DOUT a keze.

A fényképen az összekapcsolt modulokat mutatjuk be. A modul HT címe 24, azaz mivel kimeneti egység, HT-n OUT 24 utasítással szólítható meg. A többi gépen a szükséges X címkorrekciót figyelembe véve vagy OUT X+24,SZÁM, vagy POKE X+24, SZÁM a megfelelő.

Természetesen a SZÁM meghatározásánál figyelembe kell venni azt a DIN-nel kapcsolatban már tárgyalt tulajdonságot,



hogy a kimeneti pontok a megfelelő bit 1 értékére aktivizálódnak, és addig vannak aktív állapotban, míg a bit értéke 0 nem lesz.

Nézzünk egy példát! A POKE 65048,129 parancs kiadásával (C16 vagy C Plus/4) a két szélső kimenet lesz aktív. Ez könnyen kiszámolható: 129=128+1, azaz 129 (dec.)=10000001 (bin). A kimenetekre egy-egy LED-et kötve, a két szélső állapota a parancs hatására megváltozik.

```
1 PRINT"J":X=65032
10 PRINT:PRINT" DOUT TEST "
20 GET U$: IF U$="" THEN 20
30 PRINT"J"
40 POKE X+24,0
50 A=PEEK(X+24):U$=""
60 IF A>0 THEN A=5
70 PRINT:PRINT:PRINT
80 PRINT" EZ A KIMENET AKTIV";
90 PRINTA"V-OS!"
100 GOTO 20
```

1. lista

```
    ********************************
    *                              *
    *   ADÓ PROGRAM C 16 GÉPRE     *
    *                              *
    ********************************
10 REM KEPERNYÓ TÖRLÉS
20 M=400 : REM VÁRAKOZÁS HOSSZA

   * LEADANDÓ INFORMÁCIÓ *

30 A$=""
40 GET A$ : IF A$="" THEN 40
50 PRINT A$; : A=ASC(A$)
60 FOR I=0 TO 7 : A(I)=(A AND 2^I)
70 NEXT

   * ADÓ CIKLUS *

80 FOR K=7 TO 0 STEP -1
90 IF A(K)<>0 THEN J=1 : ELSE J=0
100 GOSUB 200 : REM SZINKRON
110 POKE 65048,J*129
120 FOR T=0 TO M : NEXT
130 NEXT : POKE 65048,0
140 GOTO 30
```

```
   * SZINKRONIZÁCIÓ LEFUTÓ ÉLRE *

200 POKE 65048,128
210 FOR E=0 TO 20 : NEXT
220 POKE 65048,0
230 FOR E=0 TO 20 : NEXT
240 RETURN
```

2. lista

3. lista

```
    ********************************
    *                              *
    *   VEVÖ PROGRAM C+4 GÉPRE     *
    *                              *
    ********************************
10 REM KEPERNYÓ TÖRLÉS
20 M=200 : REM VÁRAKOZÁS HOSSZA

   * VEVÖ CIKLUS *

30 FOR I=7 TO 0 STEP -1
40 GOSUB 500 : REM SZINKRONIZÁCIÓ
50 FOR K=0 TO M : NEXT K
60 A(I)=PEEK(65048) : NEXT I

   * A CHR$ KÓDOK MEGHATÁROZÁSA *

70 B=0 : FOR I= 7 TO 0 STEP -1
80 B=A(I)*2^I+B : NEXT I

   * KIÍRÁS *

90 C$=CHR$(B) : PRINT C$;
100 GOTO 30 : REM ÚJ VÉTELRE KÉSZ

   * SZINKRONIZÁCIÓ LEFUTÓ ÉLRE *

500 IF PEEK(65048)<>1 THEN 500
510 IF PEEK(65048)<>0 THEN 510
520 RETURN
```

A 8 kimeneti pont mindegyike TTL szintű (Ø V, illetve 5 V) jelet ad, amit LED-hez közvetlenül is kapcsolhatunk. Az elég nagy terhelhetőség (50 mA) arra is lehetőséget biztosít, hogy akár kisebb teljesítményű villanymotort működtessen a modul. A határt a számítógép tápegysége szabja meg. Ez sem végső korlát, van erre is megoldás; a későbbi részekben még foglalkozunk vele.

A modul kétféle kimenetet tartalmazhat. Az egyik aktív 5 V-os, a másik aktív Ø V-os pont. Az előbbi kapcsoló tranzisztorral, az utóbbi szilárdtest relével van felépítve.

Az előzőleg leírt példában a LED-ek viselkedése alapján eldönthető, hogy melyik változat került a kezünkbe. Az a kimenet relés, azaz aktív Ø V-os, amelyikhez csatlakoztatott LED a kimenet aktivizálása előtt világít, utána viszont elalszik. A fordított viselkedésű kimenet az aktív 5 V-os.

Az *1. lista* egy igen egyszerű teszt. C64-re készült, de az X változó (címkorrekció) és a GET utasítás megfelelő módosításával bármely gépre jó. (A Z80 alapú számítógépeken a PEEK helyett INP, a POKE helyett OUT kell!) Kössük a DIN és a DOUT modult a számítógéphez, majd a vizsgálandó kimenetet (DOUT) kössük bármelyik bemenetre (DIN). Egy billentyű megnyomására megkapjuk a választ.

A DOUT modul felhasználása igen sokrétű lehet, de a változatok között nem a programozási különbségek az elsődlegesek, inkább a modulhoz kapcsolt beavatkozók adják a változatosságot. Az előző példa alapján bárki bátran vállalkozhat saját ötletének megvalósítására.

Végül egy igen látványos alkalmazás: két számítógép közötti átvitel. Az adó egy DOUT modulhoz kapcsolt infra-LED (C Plus/4), a vevő egy DIN-hez kapcsolt fényelem (C16). Természetesen megvalósítható a dolog például egy egyszerű optokapuval is. Vigyázni kell a programozásnál, hogy a DOUT aktív 5 V-os kimenetét használjuk (ez nálunk a jobb szélső volt; bin. 10000000 = dec.128). Az adó programja a *2. listán*, a vevő a *3. listán* látható.

Jó munkát kívánunk mindenkinek, és várjuk a leveleket az ötletekkel, tapasztalatokkal.

ALBU LÁSZLÓ—KIRÁLY LÁSZLÓ

Kedves továbbtanulók!

Következő számunkban részletesen ismertetjük, hogy a számítástechnikával kapcsolatos tárgyakat hol, melyik iskolában oktatják az országban. Tájékoztatónk jövendő szakmunkástanulóknak, gimnazistáknak, szakközépiskolásoknak és a felsőfokú tanintézetekbe jelentkezőknek szól.

A graftálnövények legegyszerűbb változatait egy közönséges papírlap és ceruza segítségével is életre kelthetjük, igazi termőterületük azonban a számítógépeinkhez kapcsolt tévékészülékek képernyői. Száraik és leveleik elágazási mintázata a valódi növényekéhez hasonló, grafikus programozók álmodják meg őket. Néhány ilyen graftálnövényt szemléltet az *1. ábra*.

A graftálnövény elnevezés a „gráf" és a „fraktál" szavak összevonásából származik. Ugyanis e növények szerkezete a gráfokon alapul és fraktál természetük van; ez utóbbin azt értjük, hogy a tört dimenziós geometriai alakzatok közé tartoznak. Például a *2. ábrán* levő graftálfák területe nemzedékről nemzedékre megháromszorozódik, ugyanakkor magasságuk megduplázódik. Ezért a graftálfák dimenziója

$$d = \frac{\log 3}{\log 2} = 1{,}5849\ldots$$

A GRAFTAL programot C64 számítógépre írtam, és a Simon's BASIC választását az magyarázza, hogy a finomgrafika műveleteit kényelmesen programozhattam.

A graftálnövények szerkezetét megadó gráfokat valahogyan tárolnunk kell számítógépünkben. E célból a gráfok leírására egy kódrendszert vezetünk be: a gráf kódja bizonyos egyezményes jelek lineáris sorozata lesz. Ezt a jelsorozatot aztán kényelmesen bevihetjük a gépbe. Vegyük például a 2. ábrán szemléltetett graftálfákat. Ezekben háromféle szerkezeti elemet különböztethetünk meg: levéllel végződő ágacskákat, ágak és a törzs közbenső szakaszait, valamint elágazási pontokat. Kódolásnál a leveles ágacskáknak a Z, a közbenső ágszakaszoknak pedig a G szimbólumot feleltetjük meg. Az elágazási pontok szimbólumai a zárójelek: ezeket úgy alkalmazzuk, hogy a zárójelbe mindig a leágazó részfa kódja kerül. A 2. ábra első három graftálfájának kódja:

Z
G(Z)G(Z)Z
GG(G(Z)G(Z)Z)GG(G(Z)G(Z)Z)G(Z)G(Z)Z

A negyedik graftálfa kódjának felírását az olvasóra bízom.

Ez a kódolás a graftálnövényeknek csak a vázlatos felépítését írja le.

Lényeges dolog, hogy egy graftálfa kódjából egy szabályrendszer segítségével mindig levezethetjük a következő nemzedékhez tartozó fa kódját! Ezt a szabályrendszert L-nyelvtannak nevezik. Megalkotója Lindenmeyer dán biológus és matematikus volt. Az L-nyelvtan a következő szabályokból áll:

Z → G(Z)G(Z)Z
G → GG
(→ (
) →)

A nyilak azt mutatják, hogy a kód egyes szimbólumait a következő nemzedékben milyen jelsorozat fogja helyettesíteni. Könnyen meggyőződhetünk arról, hogy az L-nyelvtan — ha egyetlen Z-ből álló kódból indulunk ki — egymás után megadja a 2. ábra graftálfáinak kódjait.

Némi matematikai vizsgálódással az L-nyelvtanból a graftálfák kódjának sok érdekes tulajdonságát vezethetjük le. Az egyik az, hogy az N-edik graftálfa kódjának hosszát — a kódban levő szimbólumok számát — a következő képlet adja meg:

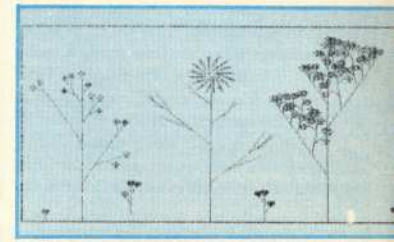
$$h_N = 5 \cdot 3^{N-1} - 2^N - 2$$

A másik a fa magassága. A graftálfa magasságán a törzsében levő egyenesszakaszok számát értjük, és ez azonos azon Z és G szimbólumok számával, melyek semmiféle zárójelbe sincsenek bezárva. A magasságot megadó képlet:

$$m_N = 2^N - 1$$

Az egyszerűbb számítógépek finomgrafikája a képernyő magasságát 200 képpontra osztja fel. Ha kikötjük, hogy a graftálfa minden ágacskája legalább három képpontnyi legyen, akkor a fa törzse legfeljebb 200/3 ≈ 66 szakaszból állhat, azaz $m_N \leq 66$. Ebből rögtön következik, hogy $N \leq 6$ és $h_N \leq 1149$. Ilyen képernyőfelbontás esetén tehát legfeljebb 1149 szimbólumot tartalmazó kódokkal fogunk találkozni.

Graftál

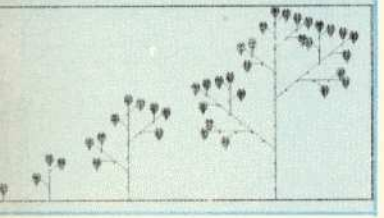


1. ábra

```
 1 REM **********************
 2 REM * GRAFTAL NOVENYEK *
 3 REM **********************
 10 DIM AB(1,1150),V(3,5)
 20 F$="F-KOV.NEMZEDEK:E-PROG.BEFEJEZES:"
 21 F$=F$+"D-NYOMTATAS"
 22 B$="BETELT A VEREM"
 23 U$="URES A VEREM"
 30 AB(0,1)=0:AB(0,2)=4
 40 K=0
 50 N=1
 60 LOOP
 70 IF N<5 THEN M=10:ELSE:M=21-3*N
 80 EXEC FIGUR
 90 GET A$
 100 EXIT IF A$="E"
 110 IF A$="D" THEN EXEC PLOT
 120 IF A$<>"F" THEN 90
 130 N=N+1
 140 EXIT IF N>6
 150 K1=K:K=1-K
 160 EXEC GENER
 170 END LOOP
 180 END
 1000 REM UJ NOVENYGENERACIO LETREHOZASA
 1001 REM ******************************
 1002 PROC GENER
 1010 I=1:J=1
 1020 LOOP
 1030 NXT=AB(K1,I)
 1040 EXIT IF NXT>=4
 1050 IF NXT=0 THEN 1100
 1060 IF NXT=1 THEN 1210
 1070 AB(K,J)=NXT
 1080 J=J+1
 1090 GOTO 1240
 1100 AB(K,J)=1
 1110 AB(K,J+1)=2
 1120 AB(K,J+2)=0
 1130 AB(K,J+3)=3
 1140 AB(K,J+4)=1
 1150 AB(K,J+5)=2
 1160 AB(K,J+6)=0
 1170 AB(K,J+7)=3
 1180 AB(K,J+8)=0
 1190 J=J+9
 1200 GOTO 1240
 1210 AB(K,J)=1
 1220 AB(K,J+1)=1
 1230 J=J+2
 1240 I=I+1
 1250 END LOOP
 1260 AB(K,J)=4
 1270 END PROC
```

vények

A GRAFTAL program egy rövid főprogramból és két fontos szubrutinból áll. A főprogram maga egyszerű ciklus, melyek N-edik lépésében a képernyőre rajzolódik az N-edik graftálnövény, majd ezt követően, az L-nyelvtan szabályai szerint létrejön az N+1-edik növény kódja. A rajzolást a FIGUR, a kódgenerálást a GENER elnevezésű szubrutin végzi.



ábra

```
2000 REM A GRAFTALNOVENY KIRAJZOLASA
2001 REM ****************************
2002 PROC FIGUR
2010 X=150:Y=199:D=0:L=1
2020 HIRES 3,9
2030 I=1:P=0
2040 LOOP
2050 NXT=AB(K,I)
2060 EXIT IF NXT>=4
2070 IF NXT=2 THEN 2140
2080 IF NXT=3 THEN 2230
2090 X1=X+M*SIN(D)
2100 Y1=Y-M*COS(D)
2110 LINE X,Y,X1,Y1,1
2120 X=X1:Y=Y1
2125 IF NXT=0 THEN CHAR X-4,Y-3,83,1,1
2130 GOTO 2270
2140 IF RND(1)<.95-.05*P+.1*L THEN 2170
2150 EXEC SKIP
2160 GOTO 2270
2170 P=P+1
2180 IF P>=6 THEN PRINT B$:END
2190 V(0,P)=X:V(1,P)=Y:V(2,P)=D:V(3,P)=L
2200 AN=*(P+3)/24
2210 D=D+L*AN
2220 GOTO 2270
2230 IF P<=0 THEN PRINT U$:END
2240 X=V(0,P):Y=V(1,P):D=V(2,P):L=V(3,P)
2250 P=P-1
2260 L=-L
2270 I=I+1
2280 END LOOP
2290 TEXT 0,0,F$,1,1,7
2300 END PROC
3000 REM NOVENYAG KIHAGYASA
3001 REM ******************
3002 PROC SKIP
3010 B=0
3020 LOOP
3030 IF AB(K,I)=2 THEN B=B+1
3040 IF AB(K,I)=3 THEN B=B-1
3050 EXIT IF B=0 OR AB(K,I+1)=4
3060 I=I+1
3070 END LOOP
3080 END PROC
4000 REM A GRAFTALNOVENY KINYOMTATASA
4001 REM *****************************
4002 PROC PLOT
4010 TEXT 0,0,F$,0,1,7
4020 COPY
4030 TEXT 0,0,F$,1,1,7
4040 END PROC
```

A rajzolás befejezése és az új kód generálásának megkezdése között a program megáll, és a gépkezelő beavatkozását várja. A program továbbfut az F betű leütésével; a D betű lenyomásával a képernyőn lévő növény képét kinyomtathatjuk, ha van mátrixnyomtatónk, az E betű lenyomása pedig a program befejezését jelenti.

A GRAFTAL program egyik legfontosabb eleme az AB nevű tömb. Ennek két oszlopa és 1150 sora van; az egyes növénynemzedékek kódjai váltakozva az AB két oszlopának egyikét foglalják el. Az egyik oszlopban lévő nemzedéket a másik oszlopban lévő, előző nemzedékből származtatjuk. Az AB tömbben célszerűen nem a Z, G, (és) szimbólumokat tároljuk, hanem helyettük egyszerűen a 0, 1, 2 és 3 számokat. Hogy a szimbólumsorozatok hosszát ne kelljen folyton kiszámolni, bevezetünk egy további szimbólumszámot, amely a szimbólumsorozat végét jelzi. Végjelzésül a 4-es számot választjuk, tehát a graftálfa kódjának feldolgozásánál addig kell az AB tömb megfelelő oszlopában haladni, amíg el nem érjük a 4-es értékű tömbelemet.

Már láttuk korábban, hogy ha számítógépünknek nincs különösen finom grafikája, akkor csak az $N \leq 6$ növénygenerációkat jeleníthetjük meg a képernyőn. Ebből következik, hogy a végjellel kiegészített kód hossza legfeljebb $h_6 + 1 = 1150$ lehet, és ez az oka, hogy az AB tömbnek éppen 1150 sorának kell lennie.

A GENER szubrutin az L-nyelvtan szabályait testesíti meg. Az előző graftálnövény kódja az AB tömb K1 indexű oszlopában van, az új nemzedék kódja a K-val indexelt oszlopba kerül. Az oszlopok váltogatását a főprogram 150. utasítása végzi. Az új nemzedék kódolása akkor ér véget, amikor a K1 oszlopban elérjük a végjelzést; ezt az 1040. utasítás figyeli. Az új kód végére a végjelet az 1260. utasítás írja.

A kirajzolást végző FIGUR szubrutin az AB tömb K indexű oszlopát dolgozza fel. E rutinnak négy alapvetően fontos változóját jelöli X, Y, D és L. X és Y tulajdonképpen egy képernyőpont koordinátái: azt a pontot határozzák meg, ahová a növény ágazatának kirajzolásánál pillanatnyilag eljutottunk. D egy szögváltozó: az éppen kirajzolandó ágszakasz irányát adja meg. (D értéke radiánokban értendő.) Az L változó értéke csak ± 1 lehet: ez azt határozza meg, hogy a növény rajzolatának következő elágazási pontjánál jobbra vagy balra forduljon-e, azaz D értéke növekedjen vagy csökkenjen. Ez a négy változó a FIGUR működése közben folyton aktualizálódik. Induló értékeiket a 2010. utasítás állítja be. Itt X és Y a növény talppontjának koordinátái (ezek értékét módosítva a növényt áthelyezhetjük a képernyő bármely pontjára). Az irányszög D=0 kezdőértéke azt jelenti, hogy a növény függőlegesen felfelé kezd növekedni, és L=1 azt, hogy az első gally jobbra ágazik.

A FIGUR szubrutin is tulajdonképpen egyetlen ciklusból áll, amely végigfut a növény kódjának szimbólumain. A ciklusmag egy három részre bomló elágazásból áll.

1. Ha a soron következő szimbólumszám értéke 0 vagy 1, akkor a graftálnövény törzsének vagy valamelyik ágának egy szakaszát kell a képernyőre rajzolni. A szakasz kezdőpontját az X és Y változók határozzák meg, irányát D, hosszát pedig az M változó adja meg. (M értéke egy növényen belül sohasem módosul.) A szakasz rajzolását és X, Y aktualizálását a 2090—2120. utasítások végzik el.

Ha a szimbólumszám értéke zérus, akkor ágvégződésről van szó. Ilyenkor az ág végére valamilyen levélkét vagy virágot rajzolunk. Erre szolgál a program 2125. utasítása.

2. Ha a szimbólumszám értéke 2, akkor a kódban egy nyitó zárójelhez érkeztünk, ami egy elágazási pontot jelent. Ilyenkor az X, Y, D és L változók pillanatnyi értékeit kimentjük egy veremtárolóba, hogy majd a zárójel bezárásakor elővehessük ismét őket. A vermelést a 2170—2190. utasítások testesítik meg. Ezekben P a veremtár mutatója, és ennek értéke egyben jelzi nekünk azt, hogy a képernyőn éppen hányadrendű ágat rajzolunk: a törzsnél P=0, a törzsről kiágazó főágaknál P=1, a hatodik növénynemzedék legapróbb ágacskáinál P=5.

X, Y, D és L értékeinek tárolása után az elágazási pontban megváltoztatjuk a D irányszög értékét. A változás nagyságát az AN változó tartalmazza, előjelét pedig L; D változtatására a 2210. utasítás szolgál.

3. Ha a szimbólumszám értéke 3, akkor a kód feldolgozásában egy záró zárójelhez jutottunk el. Ilyenkor folytatnunk kell az elágazási pontban abbahagyott ág rajzolását. E célból a verem tetejéről le kell emelnünk X, Y, D és L értékeket (2230—2250. utasítások), hiszen ezek adják meg az abbahagyott ág adatait. Ugyanekkor megváltoztatjuk L előjelét, hogy a következő elágazási pontban majd a másik oldalra ágazzon a gally (2260. utasítás).

A FIGUR szubrutinunk a legegyszerűbb növényrajzoló eljáráshoz képest két finom bővítést tartalmaz. Az egyik, hogy az elágazási pontokban az ágak hajlásszöge nem egy adott állandó érték, tehát nem olyan, mint a 2. ábrán, ahol ez a szög mindig 45°. A FIGUR szubrutinban az elágazási szög értékét a 2200. utasítás állítja be, és ez a szög növekvő P-vel növekedni fog: a törzsről a főágak (P=1 eset) elég meredeken, 30°-os szögben törnek ki, míg a P=5 esetnek megfelelő gallyacskáknál az elágazási szög már 60° lesz.

A másik finomítás, hogy ha a graftálnövény kódjában egy elágazási ponthoz érkezünk, akkor az oldalágat csak bizonyos valószínűséggel rajzoljuk meg. A valószínűségi formula a 2140. utasításban van. Látjuk, hogy az oldalág kirajzolásának valószínűsége növekvő P-vel csökken: a főágakat még nagy valószínűséggel kirajzoltatjuk, a kisebbeket már nem. Ráadásul formulánkat az L változó is befolyásolja: a jobbra történő kiágazásnak nagyobb a valószínűsége, mint a bal oldalinak, tehát a graftálnövényünk aszimmetrikus lesz. Ha az RND-vel generált véletlenszám értéke történetesen az oldalág lemaradásának felel meg, akkor az SKIP nevű szubrutint kell behívnunk. Ez a szubrutin semmi mást nem csinál, csak megkeresi a kódban az elágazási pontnak megfelelő záró zárójelet, és a kód feldolgozása innen fog folytatódni.

A lelkesebb „diszkertészek" próbálják meg az itt közölt programot néhány ponton módosítani! Például a program 2125. utasítása minden ágvégződésre ugyanazt a levelecskét rajzolja: egy kis szív alakú karaktert, melynek képernyőkódja 83. Megtehetjük, hogy a különböző ágvégződésekre két vagy három különböző levél és virág közül választjuk ki valamelyiket; a választást rábízhatjuk véletlenszám- vagy valamilyen más algoritmusra. Ezt tettük az 1. ábra bal oldali graftálnövényénél is. Némi fáradsággal mutatós virágot kibontó szubrutint is készíthetünk, mint amilyet például az 1. ábra középső növényének csúcsánál alkalmaztunk.

A program 2200. utasításában módosíthatjuk az ágak hajlásszögét megadó formulát. Például az 1. ábra jobb szélső növényénél elértük, hogy a törzsről kiágazó főágakból minden gally felfelé növekedjék. A 2140. utasítás módosításával tetszőlegesen megváltoztathatjuk az oldalág kihajtásának valószínűségét. Ezt a valószínűséget nemcsak P-től és L-től tehetjük függővé, hanem N-től is, tehát attól, hogy hányadik növénynemzedékről van szó. Ha a különböző variációkat ugyanarra a képernyőre rajzoljuk fel, akkor fákkal, cserjékkel és virágokkal borított fantasztikus parkokat varázsolhatunk magunk elé.

A fenti módosítások mind a FIGUR szubrutinra vonatkoznak, és nem érintik az AB tömb tartalmát. Más szóval a különböző növényvariációk csak abban térnek el egymástól, hogy másként értelmezik a növény kódját. A graftálnövények fajainak milliárdjait teremthetjük meg, ha magának az L-nyelvtannak a szabályait is megváltoztatjuk!

KISDI DÁVID

LIST parancs titkosítása C64-re

Az alábbi programnak nem feladata, hogy letiltsa a LIST utasítást, az viszont igen, hogy egy bizonyos kulcsszótól függővé tegye a listázást.

A betöltés után a programot SYS 49152-vel indíthatjuk el, majd a LIST parancs után a gép a PASSWORD? üzenetet írja ki és megjelenik a kurzor. Ha ezt követően beírjuk a jelszót és lenyomjuk a <RETURN> gombot, akkor megkapjuk a programlistát. Ám, ha a jelszó nem volt jó, akkor semmi sem történik, illetve a READY üzenetet kapjuk. És amikor egy vállalkozó kedvű számítógépes benyúl a programba — ezt ugyanis megpróbálhatja —, valószínűleg nem sokra megy, mert a C000-tól C0FF-ig tartó területre nem érvényes a POKE utasítás. A gép ilyenkor a CAN'T POKE ERROR-t írja ki. A program részletes magyarázata az assembler listában található. Aki pedig a BASIC programmal kíván dolgozni, az a futtatás után, a gép MI A JELSZÓ? kérdésére adja be a jelszót, s ettől kezdve a gép kikapcsolásáig csak ezzel a jelszóval listázhat.

KORSÓS ISTVÁN

```
100 REM *** LIST TITKOSITAS ***
110 DATA 169,011,141,008,003,169,192,141
120 DATA 009,003,096,032,115,000,201,151
130 DATA 240,010,201,155,240,038,032,121
140 DATA 000,076,231,167,032,115,000,032
150 DATA 235,183,165,021,201,192,240,008
160 DATA 138,160,000,145,020,076,174,167
170 DATA 169,109,160,192,032,030,171,162
180 DATA 128,108,000,003,169,096,160,192
190 DATA 032,030,171,162,000,032,207,255
200 DATA 221,129,192,208,014,201,013,240
210 DATA 004,232,076,069,192,032,121,000
220 DATA 076,231,167,162,128,160,000,003
230 DATA 013,080,065,083,083,087,079,082
240 DATA 068,032,063,032,000,013,063,032
250 DATA 067,065,079,039,084,032,080,079
260 DATA 075,069,032,069,082,082,079,082
270 DATA 000,999
280 C=0
290 FOR X=49152 TO 49290:READ A
300 IF A=999 THEN 320
310 POKE X,A:C=C+A:NEXT
320 IF C=12814 THEN 340
330 PRINT "ADAT-HIBA VAN !!":END
340 INPUT "MI LEGYEN A JELSZO: ";A$
350 L=LEN(A$):Y=49280+L
360 FOR X=49281 TO Y
370 POKE X,ASC(MID$(A$,X-49280,1))
380 NEXT
390 POKE X,13:X=X+1:POKE X,0
400 SYS 49152:PRINT "2":NEW:REM CLR
```

```
100 SYS#4096:.OPT OO
110 *=49152
120 ;
130 LDA #<NEW:STA $308
140 LDA #>NEW:STA $309:RTS
150 ;
160 NEW JSR $73:CMP #151:BEQ POKE
170 CMP #155:BEQ LIST
180 JSR $79:JMP $A7E7
190 ;
200 POKE JSR $73:JSR $B7EB
210 LDA $15:CMP ##C0:BEQ EL
220 TXA:LDY #0:STA (#14),Y:JMP $A7AE
230 EL LDA #<ERRR:LDY #>ERRR:JSR $AB1E:LDX ##80:JMP (#300)
240 ;
250 LIST LDA #<MESS:LDY #>MESS:JSR $AB1E
260 LDX #0
270 CIKLUS JSR $FFCF
280 CMP PASS,X:BNE NO
290 CMP #13:BEQ YES:INX:JMP CIKLUS
300 YES JSR $79:JMP $A7E7
310 NO LDX ##80:JMP (#300)
320 ;
330 MESS .BYTE 13:.ASC "PASSWORD ? ":.BYTE 0
340 PASS .ASC "OCTO":.BYTE 13:0
350 ERRR .BYTE13 .ASC "? CAN'T POKE ERROR":.BYTE 0
```

A beszélő Spectrum

A mellékelt program a ZX-Spectrum számítógép magnóbemenetére érkező jelet digitalizálva — de kissé torzítva — tárolja, majd visszajátssza. Ehhez mindössze egy mikrofont kell a géphez csatlakoztatni és a jeleket, így akár a beszédet is, megfelelő szinten bejátszani. A program 48 k-s gépen fut és a teljes memóriát felhasználja — még a képernyő memóriáját is. (Ezért az ott tapasztalható zagyvaságok a természetes működést jelentik.) A módszerrel mintegy 5-6 másodpercnyi anyagot vehetünk fel.

A program a következő billentyűknek engedelmeskedik:

— ENTER: felvétel indul
— P: lejátszás indul
— SPACE: lejátszás leállítása
— Q: hidegstart

A felvételt három sípszó előzi meg, a lejátszás folyamatosan ismétlődik.

A programot a 65101-es címre kell fordítani és tölteni. Futtatása előtt érdemes a forrásprogramot kazettára menteni, hogy

```
Az assembler list:

        ORG 65101
BILL    DI
        LD SP,65535
B1      LD A,251        ;quit
        IN A,(254)
        RRA
        JP NC,0
        LD A,191        ;record
        IN A,(254)
        RRA
        CALL NC,BRE
        LD A,223        ;play
        IN A,(254)
        RRA
        CALL NC,CONT
        JR B1
CONT    CALL PLAY
        LD A,127        ;break
        IN A,(254)
        RRA
        RET NC
        JR CONT
BRE     CALL BEEP
        CALL REC
        RET
BEEP    LD DE,110
        LD HL,964
        CALL 949
        CALL DEL
        LD DE,110
        LD HL,964
        CALL 949
        CALL DEL
        LD DE,110
        LD HL,964
        CALL 949
        CALL DEL
        RET
DEL     LD BC,30000
C1      DEC BC
        LD A,B
```

```
        OR C
        JR NZ,C1
        RET
REC     DI
        LD HL,16384
        LD DE,48717
L1      LD B,7
        LD C,0
L2      IN A,(254)
        AND 64
        ADD A,C
        RLCA
        LD C,A
        DJNZ L2
        LD A,C
        LD (HL),A
        INC HL
        DEC DE
        LD A,D
        OR E
        JR NZ,L1
        LD DE,130
        LD HL,1642
        CALL 949
        LD HL,16384
        LD DE,48717
L3      LD A,(HL)
        RRCA
        LD (HL),A
        INC HL
        DEC DE
        LD A,D
        OR E
        JR NZ,L3
        RET
PLAY    DI
        LD HL,16384
        LD DE,48717
L4      LD B,7
L5      LD A,(HL)
        OUT (254),A
        RLCA
        NOP
        NOP
        NOP
        DJNZ L5
        INC HL
        DEC DE
        LD A,D
        OR E
        JR NZ,L4
END     RET
```

ha hiba kerülne netán a munkába, ne veszszen kárba a hosszú munka.

Ajánlatos továbbá a programot PAUSE0:RANDOMIZE USR 65*01* utasítással, majd egy ENTER-től különböző gomb lenyomásával indítani, mert az ENTER azonnali felvétellel járhat.

CSAJTAI KORNÉL

Bővített BASIC HT-re

A következő assembler lista négy új, bővített utasítást tartalmaz. A POKE x, y, z . . . az X címtől kezdődően sorra elhelyezi az y, z stb. adatokat.

Az új GOTO és GOSUB utasítás konkrét sorszám helyett kifejezést is elfogad. A PUT utasítás a DATA pointert állítja a kifejezés értéke által meghatározott sorszámra.

A programot az EDITOR segítségével írjuk be, fordítsuk le, ezt követően pedig vegyük fel. Betöltés után a program elindul, letiltja a memória végét, majd visszatér a beolvasási főágba.

KÖVESI ZSOLT

```
;  *** BOVITETT BASIC ***
;
;  KESZITETTE: KOVESI ZSOLT
;
;  POKE X,Y,Z,...,V,...  ARGUMENTUM
;  GOSUB X (ARGUMENTUM)
;  GOTO  X (ARGUMENTUM)
;  PUT   X (ARGUMENTUM): A DATAPOINTERT
;                        BEALLITJA
ORG 41E2H
JP 7F00H            41E2H C3007F
ORG 4003H
JP 7F15H            4003H C3157F
ORG 7EFBH
POP HL              7EFBH E1
LD HL,7EFEH         7EFEH 21FE7E
LD (40B1H),HL       7F01H 22B140
LD HL,7ECCH         7F04H 21CC7E
LD (4080H),HL       7F07H 228040
LD A,0C9H           7F0AH 3EC9
LD (41E2H),A        7F0CH 32E241
CALL 1E5DH          7F0FH CD5D1E
JP 1B19H            7F12H C3191B
EXX                 7F15H D9
POP HL              7F16H E1
LD DE,1D5BH         7F17H 115B1D
RST 18H             7F1AH DF
PUSH HL             7F1BH E5
EXX                 7F1CH D9
JP NZ,1D78H         7F1DH C2781D
RST 10H             7F20H D7
RET Z               7F21H C8
LD A,(HL)           7F22H 7E
CP 0B1H             7F23H FEB1
JR NZ,GOTO          7F25H 2010
RST 10H             7F27H D7
CALL 20B1H          7F28H CDB120
VISSZA: DEC HL      7F2BH 2B
RST 10H             7F2CH D7
RET Z               7F2DH C8
INC DE              7F2EH 13
LD BC,VISSZA        7F2FH 012B7F
PUSH BC             7F32H C5
PUSH DE             7F33H D5
JP 2CB5H            7F34H C3B52C
GOTO:CP 8DH         7F37H FE8D
JR NZ,GOSUB         7F39H 2008
POP BC              7F3BH C1
GO:LD BC,1D1EH      7F3CH 011E1D
PUSH BC             7F3FH C5
CALL 2B01H          7F40H CD012B
JP 1FC5H            7F43H C3C51F
GOSUB:CP 91H        7F46H FE91
JR NZ,PUT           7F48H 2012
LD C,3              7F4AH 0E03
CALL 1963H          7F4CH CD6319
POP BC              7F4FH C1
PUSH HL             7F50H E5
PUSH HL             7F51H E5
LD HL,(40A2H)       7F52H 2AA240
EX (SP),HL          7F55H E3
LD A,91H            7F56H 3E91
PUSH AF             7F58H F5
INC SP              7F59H 33
JR GO               7F5AH 18E0
PUT:CP 0A5H         7F5CH FEA5
JR NZ,VEGE          7F5EH 2015
CALL 2B01H          7F60H CD012B
PUSH HL             7F63H E5
CALL 182CH          7F64H CD2C18
DEC BC              7F67H 0B
POP HL              7F68H E1
LD (40FFH),BC       7F69H ED43FF40
LD (40D4H),DE       7F6DH ED53D440
POP BC              7F71H C1
JP 1B1EH            7F72H C31E1D
VEGE:RET            7F75H C9
```

Rövid játék VIC—20-ra

Egy üldöző elől kell menekülni a programban. Cél, hogy minél többet lépjünk anélkül, hogy elkapjon az ellenfél.

vÁltozók:

A: a játékos X pozíciója
B: a játékos Y pozíciója
C: az üldöző X pozíciója
D: az üldöző Y pozíciója
I: (villogásmentes törlés)
a játékos előző pozíciója
J: az üldöző előző pozíciója
T: a lépések száma
K: az előző lépések száma

Irányítás:

E—fel, X—le, S—balra, D—jobbra, $—sztring (zárójel, értelem szerint nyitó, záró), 0—nulla.

HAJNAL CSABA

```
0 DEFFNA/A/=INT/RND/TI/#22/:POKE65o,128
1 DEFFNB/B/=768o+22*B+A
2 DEFFNC/C/=768o+22*D+C
1o PRINTCHR$/147/,CHR$/5/:POKE36879,42
2o A=FNA/A/:B=FNA/A/
3o C=FNA/A/:D=FNA/A/
4o T=-1:IFA=CORB=DGOTO2o
5o IFK$="D"THENA=A+1:IFA>21THENA=21
6o IFK$="S"THENA=A-1:IFA<oTHENA=o
7o IFK$="X"THENB=B+1:IFB>21THENB=21
8o IFK$="E"THENB=B-1:IFB<oTHENB=o
9o POKEFNB/B/,83
1oo IFFNB/B/<>ITHENPOKEI,96:I=FNB/B/:
   T=T+1
11o IFA>CTHENC=C+1
12o IFA<CTHENC=C-1
13o IFB>DTHEND=D+1
14o IFB<DTHEND=D-1
15o POKEFNC/C/,86
16o IFFNC/C/<>JTHENPOKEJ,96:J=FNC/C/
17o IFA=CANDB=DGOTO2oo
18o FORZ=otO1oo:NEXT
19o GETK$:GOTO5o
2oo PRINT T"LEPES"K"UTAN
21o K=T:FORZ=oTO2E3:NEXT
22o GOTO1o
```

Kapcsoltam? – Kapcsoltam! – a C64-en

Népszerű a Kapcsoltam című televíziós műsor. Valószínű, hogy sokan szeretnének válaszolni a néha igencsak egyszerű, máskor meg furfangosan nehéz számsorozatoket felvonultató kérdésekre. (Az egyik lehetséges választ lásd 1987/2. számunkban. — A szerk.)

A megoldáshoz azonban tudnunk kell, hogy bármilyen számsorozatot veszünk, annak nemcsak egy, hanem számos folytatása lehetséges. Ezt felismerve a feladat elkészítése már igen egyszerű. (Az általános képlet az ábrán olvasható.)

Vegyünk egy példát. A sorozat első három eleme az 1; 2; és a 3. A folytatás látszólag egyszerű, vagyis a 4 következik. Valójában azonban nem feltétlenül csak ez az egyetlen megoldás. Ugyanis hasonló módon az 5 is elfogadható lehetne az $X(n+1)=X(n)+1+INT(X<n>3)$ képlet alapján. Természetesen minél több számot adnak meg, annál bonyolultabbá válhat az azt leíró „kényszerképlet".

A feladatot számítógépre vihetjük az alábbi programmal, amelyet logikailag — a megoldhatósága szerint — két részre oszthatunk. Ha a beírt számok reális számsort alkotnak, s ehhez mennyiségük is megfelelő, a végrehajtás az N-ed fokú számolás alapján történik. (Nem képlet!) Ha ezek a feltételek nem teljesülnek, a program a korábban leírt megoldást végrehajtó ciklushoz fordul. Természetesen a kétféle megoldás nem mindig ad azonos eredményt, így az N-ed fokú számolást a másik fölé kell rendelni, s azt csak az ily módon meg nem oldható feladatoknál szabad alkalmazni. Például az exponenciális vagy szinuszos egyenleteknél.

Figyelem! Nem mindig kapunk az eredetivel azonos megoldást. Hiszen nem is az eredeti képletet használtuk. Egyébként a programokat nemcsak a Kapcsoltam című műsorhoz alkalmazhatjuk. Jó szolgálatot tehet a különböző matematikai feladatok megoldásakor.

BÁRTFAI BARNABÁS

```
10 PRINT"■  ************************
20 PRINT"  *                      *
30 PRINT"  *     ATOMSOFT  1987   *
40 PRINT"  *                      *
50 PRINT"  *   BARTFAI BARNABAS   *
60 PRINT"  *                      *
70 PRINT"  ************************
80 PRINT"
90 PRINT
100 INPUT"HANY SZAM VAN A SORBAN";Q
105 IF Q>2 THEN 110
106 PRINT"SAJNALOM DE ENNYI SZAMMAL NEM"
107 PRINT"LEHET EGYENLETET MEGOLDANI■
108 GOTO 100
110 DIM X(Q+1,Q),Z(Q)
120 FOR I =1 TO Q
130 INPUT X(I,1)
140 NEXT
150 FOR J=2 TO Q
160 FOR I=1 TO Q-J+1
170 X(I,J)=X(I+1,J-1)-X(I,J-1)
180 NEXT:NEXT
200 I=2
210 IFX(1,I)=0THEN300
220 IF I=Q THEN 500
230 I=I+1
240 GOTO 210
300 J=2
310 IF X(J,I)<>0 THEN 220
320 IF J=Q THEN N=I-2:GOTO400
330 J=J+1
340 GOTO 310
400 FOR J=Q TO 2 STEP -1
410 X(Q-J+3,J-1)=X(Q-J+2,J-1)+X(Q-J+2,J)
420 NEXT
430 GOTO 1000
500 PRINT"■■  SAJNOS NEM ADOTT MEG ELEG"
505 PRINT"  SZAMOT, IGY KENYTELEN"
510 PRINT"  VAGYOK  NEM VALODI"
520 PRINT"  SOROZATNAK ERTELMEZNI ."
530 PRINT"■ AZ EGYENLETET  AZ ■■
540 PRINT" X(N+1)=X(N)+"
542 PRINT" +(X(2)-X(1))+INT(X(N)/X(2))*"
545 PRINT" *(X(3)-2*X(2)+X(1))+...+"
550 PRINT" +INT(X(N)/X(M-1))*(X(M-2)-"
555 PRINT" -2*X(M-1)+X(M))"
560 PRINT"■   FORMULAVAL OLDOTTAM MEG ."
600 REM
610 FOR I=2 TO Q-1
620 Z(I)=X(I-1,1)+X(I+1,1)-2*(X(I,1))
630 NEXT:S=X(1,2):I=1
640 I=I+1
650 S=S+INT(X(Q,1)/X(I,1))*Z(I)
660 H=H+1:IF I<Q-1 THEN 640
670 X(Q+1,1)=X(Q,1)+S
680 PRINT"■ IGY A KOVETKEZO TAG :";
690 PRINT X(Q+1,1)
990 END
1000 PRINT"■ A FUGGVENY"N"-ED FOKU VOLT"
1010 PRINT"■AZAMSOR KOVETKEZO ELEME :";
1015 PRINT X(Q+1,1)
1020 END
```

$$X_{n+1}= X_n+(X_2-X_1)+INT(X_n/X_2)\cdot(X_3+X_1-2X_2)+\ldots+INT(X_n/X_{m-1})\cdot(X_m+X_{m-2}-2X_{m-1})$$

X_1 a sorozat első tagja
X_n a sorozat utolsó tagja
X_{n+1} a kiszámolni kívánt szám
X_m az utolsó megadott szám

$$X_1,\ X_2,\ldots,\ X_{m-2},\ X_{m-1},\ X_m,\ \ldots,\ \overset{?}{\boxed{X_n,\ X_{n+1}}}$$

Szerkesztőtársunk mondd, akarsz-e lenni?

Vidámak, lármásak, egymást ugratók. Hangjuk le s fel rohangál. Arcukra elkelne időnként a friss pattanásokat eltüntető gyógyír. Vagyis pontosan olyanok, mint bármelyik önmagát kereső, a világot megérteni akaró kamasztársuk. Azért mégiscsak van egy csipetnyi különbség, mely időnként cseppet sem elhanyagolható. Ők már tudják, az életben mit akarnak, s tizenévesen elkötelezték magukat egy szenvedélynek, hivatásnak. És mostanság mániájukhoz társakat keresnek . . .

A Mikroszámítógép Magazin diákszerkesztőségének három tagjával válogatjuk e lapszám rovatába beküldött cikkeket. Az amúgy nyolcas társaságból a többiek éppen suliban vannak. A munkából azért nem maradtak ki, hiszen korábban együtt döntöttek a közlésre alkalmas írásokról.

Bátran állítom, bármelyik felnőtt szerkesztőségi kollektíva megirigyelheti azt az alaposságot, lelkiismeretességet, kritikus szellemet, amellyel ezek a gyerekek dolgoznak.

Bartos Gyula ifjonti rovatszerkesztő kis csapatuk céljairól mondja.

— A forgalomban lévő valamennyi géptípushoz várjuk a srácok írásait. Olyanokat, amelyek érdekelhetik az olvasókat. Ezt a rovatot a diákoknak diákok szerkesztik, nem csoda hát, ha nekik, róluk, velük, munkáikkal kívánjuk az oldalakat megtölteni. Ahhoz azonban, hogy a jelenleginél színesebb legyen a Diákrovat, mindenfajta írásnak szerepelnie kellene benne.

A változatosság viszont nem elsősorban a szerkesztőkön múlik, hanem a szerzőkön, a számítástechnikával foglalkozó társainkon. Sajnos eddig meglehetősen egyívású, egysíkú cikkeket kaptunk. Programok, programok és programok érkeznek. Helyettük, mellettük szívesen látnánk — biztosak vagyunk abban, hogy az olvasók is — minden sort, ami a tizeneveseket a számítógépek világából érdeklő, megfogó témával foglalkozik. Természetesen tudjuk, az iskola mellett csak kevesen szánják rá magukat az írásra, a fogalmazásra. Amit szerkesztőségünk segítségül igérhet — a publikált munka honorálása mellett — az az, hogy „gatyába rázzuk", vagyis közlésre alkalmassá tesszük az írásokat.

Magam sem úgy kezdtem, hogy cikkek özönével árasztottam el a szaksajtót . . . Hetedikes voltam, amikor anyukám munkahelyén a pénzügyi teendőket számítógépre vitték. A programozóval összebarátkoztam, s hétvégeken hozzá jártam Commodore 64-en tanulni. Továbbtanuláskor még nem tudtam, hogy mi legyek. Lelkesedésemet látva összeült a családi kupaktanács: „a fiút a Hámán Kató szakközépiskolába íratjuk, ott is a programozó ágazatra". Azóta nyűvöm szegény tanáraimat a szakközépben, s harmadik esztendeje tudom, hogy szüleim döntése helyes volt. Énnekem nem kell az élet útvesztőiben bukdácsolnom, határozott a célom: szoftveres leszek! A SZÁMALK hároméves tanfolyamára készülök, hogy annak végén diplomát szerezzek.

Egyébként a magazin régi olvasójaként kaptam kedvet arra, hogy írjak ide. Beküldtem néhány cikket, ezekből közölt az újság is, sőt meghívtak a Berzsenyi Gimnázium számítástechnikai klubjába. Péntek délutánonként együtt dolgozom a korombéli számítógépes megszállottakkal, s futtatjuk, ellenőrizzük a diákszerkesztőséghez érkezett programokat. Persze saját ötleteink „kikísérletezésére" is jut idő. Az összejövetelek nyitottak, bárki közénk állhat. Mindenkinek ajánlom a jó társaságot, mert hiszen, ha valaki közénk jön, az barátokra talál.

Oláh Gábor helyeslően bólogat.

— Csatlakozom az előttem szólóhoz, akivel egészen jó szakmai kapcsolatban állok. Én is ugyanabban az életkorban ismerkedtem meg a számítógéppel, de az Spectrum volt. Egy barátomé, óriási — mondhatom —, gigászi küzdelmeket folytattunk a játékprogramokkal. Szerencsére mára véget ért ez a játékszenvedély. Manapság csak programokat készítek, ezeknek viszont semmi közük a játékhoz. Autodidakta módon, magam tanultam a programozást. Könyvből ismertem meg a BASIC nyelvet, s javában írtam már kisebb-nagyobb programokat, mikor még nem is volt számítógépem. Végül szüleim megsajnáltak, s az egyik karácsonykor ajándékul egy C64-et adtak. Azt hiszem, akkor én voltam a földkerekség legboldogabb embere . . . Persze a számítógép elvont a tanulástól, egy ideig éjszaka írtam apró programjaimat, s tanultam a gépi nyelvet. Sportolás mellett eléggé lemaradtam a tanulásban, de kit érdekelt . . .? Ahogy múlt az idő, mind komolyabb programokat írtam, így készült a szövegszerkesztő szoftverem. Vagy másoktól kölcsönzött programjaim, amiket rendre sorra átírtam, továbbfejlesztettem. Felvettek a Toldi Ferenc Gimnáziumba. Szorgalmasan tanulok . . . elsősorban a legkülönbözőbb programnyelveket . . .

Készítettem egy karakterkészlet-szerkesztő programot — a Mikroszámítógép Magazinban meg is jelent —, s ezzel egyenes út vezetett a diákszerkesztőségbe. Nyáron részt vettem — társaimmal együtt — a tiszafüredi számítástechnikai táborban, s utána a hasonló jellegű lengyelországi összejövetelen.

Hogy mit várok magunktól szerkesztőktől, s ugyanakkor az olvasóktól is? Az eddigieknél még jobb, színvonalasabb írásokat. Újdonságokat bemutató, különlegességeket ismertető cikkeket. Szeretnénk megújítani a korosztályunknak szóló rovatot, de ez biztosan nem sikerül nélkülük! Jó lenne kis csapatunkat bővíteni elszánt, hozzánk hasonló őrültekkel.

Bártfai Barnabás gyorsan közbeszól.

— Nemcsak budapestieket várunk magunk közé, hanem az ország bármely pontjáról fogadjuk a jelentkezőket! A Berzsenyi Gimnáziumban a pénteki szeanszokra Soprontól Debrecenig érkeznek gyerekek. Havonta egyszer ugyan, de jönnek. Hiszem, tudom, hogy annyi kilométert nem hiába tesznek meg . . .

Szerkesztőségünkben igazi csapatmunkára vágyunk. A csapathoz azonban a mostaninál több vállalkozó kellene. Ötleteink, elképzeléseink akadnak szép számban, de szeretnénk, ha ezekhez a korunkbeli „szenvedélybetegek" is adnának muníciót. A legjobb tudásunk és képességünk szerint válaszolunk a hozzánk írt levelekre. Ugyanakkor véleményeket, kritikákat, építő jellegű bírálatokat is közreadnánk. Már ha lennének ilyenek. Természetesen nem olyat, amikor a spectrumos szidja a Commodore-t, vagy a sinclaires ócsárolja az Atarit. Az ilyen megnyilvánulásoknak persze nem lenne értelme. Annak viszont igen, ha például a lapban megjelent egyik írás nyomán valakinek „elindult" az agya, s az ott közöltek alapján valami újat alkotott. És szeretnénk, ha a gyerekek jeleznék, hogy milyen szerkesztőségi munkára vállalkoznának. Akár postai úton is! Például Kovács Béla Pusztaszabolcsból felvállalná az Enterprise gépekre hozzánk érkező programok ellenőrzését (akkor mi azokat rögvest címére küldenénk). Vagy jelentkezne valaki Kecelről, hogy a környéken ismer néhány klubot, számítógépen dolgozó kis srácot, akik megannyi újdonsággal szolgálhatnak. Felkeresné őket, s cikket írna róluk. Sorolhatnám persze a variációkat, inkább azonban a folytatást az olvasókra bízom, türelmetlenül várva, hogy mivel rukkolnak elő.

Bemutatkozásként magamról annyit, hogy már ugyan kinőttem a középiskolát — most a SZÁMALK-ban elektroműszerész vagyok —, azért remekül kijövök a diákokkal. A többieknél később találkoztam számítógéppel, igaz azóta sem szakadtunk el egymástól. Székesfehérvárott jártam elektroműszerész szakközépbe. Ott már többeknek volt számítógépe. Lázasan vetettem magam a technikába. Egyedül tanultam meg a BASIC nyelvet, majd kisebb gépeken próbálkoztam. Később az NSZK-ból kaptam egy LASER 210-es 8k-s gépet, de ehhez kevés dokumentációra leltem itthon. Felkerestem a magazint, hátha segítenek . . . Összebarátkoztunk, hívtak. Azóta szoros az együttműködésünk.

Míg Barnabással beszélgettünk, a másik két fiú egy rajzolóprogramot töltött Gábor Commodore 64-ébe. Riportalanyom egyre inkább feszengve ült a székén, félszemmel vadul a monitort követve. Megsajnáltam, elengedtem. A három srác körül megszűnt a világ. Egymást ugratva, tanácsokat osztogatva ösztökélték újabb gondolatokra.

KRASZNAI ÉVA

BASIC és gépi kód

Legutóbb olvasói észrevételekről írtam. Most a gépi kódú rutinok egyik aktivizálási módjáról, az USR függvényről lesz szó. Erről a témáról ebben a sorozatban már tettem említést az 1986/6. számban.

Az USR hasonlóan működik, mint a BASIC aritmetikai függvényei, azzal az eltéréssel, hogy míg az utóbbiak végrehajtható gépi kódja az interpreter ROM-jában megtalálható, az USR gépi kódú rutinját a felhasználó írja meg, ő helyezi el a rutint a RAM-ban és tölti a rutin belépési címét az USR vektorba.

A témával kapcsolatosan sok téves információ forog közkézen, most ezek közül veszünk szemügyre néhányat.

Dr. Úry Lászlónak az LSI ATSZ kiadásában megjelent, Commodore gépekről írt könyveiben az USR függvény ismertetésénél többek között az alábbiakat olvashatjuk:

„Szintaxis: USR (<aritmetikai kifejezés>)

Az aritmetikai kifejezés értékének a 0—65535 intervallumba kell esnie (kerekítés után)."

A C128-ról szóló kötetből további információkat kapunk: „Az interpreter a függvény argumentumának értékét az 1. lebegőpontos akkumulátorba teszi, s utána egy JSR $1218 utasítást hajt végre. A gépi kódú programból való visszatérés után az 1. lebegőpontos akkumulátor értékét két byte-os egész számmá konvertálja, és ez lesz a USR függvény értéke."

Ha az argumentum a lebegőpontos akkumulátorba kerül, mi értelme van az érték korlátozásának? És miért kellene az eredményt kétbájtos egésszé konvertálni, hiszen a Commodore gépek ROM-ban lévő interpretere szívesebben dolgozik valós, mint egész számokkal. Mintha a könyvek szerzőjének fogalma sem lenne a témáról, melyről ír! Az *1. listán* látható C64-es program és a *2. listán* lévő futási eredmény bizonyítja, hogy a fenti állítások nem felelnek meg a valóságnak.

A 100-as sorban az USR vektort egy RTS utasításra állítjuk, a hívott rutinban semmi nem történik, ezért függvényértékként magát az argumentumot kapjuk vissza. Az eredmény önmagáért beszél.

Tévedés az is, hogy a függvény paramétere csak aritmetikai kifejezés lehet. Kevesen tudják, hogy karakterlánc-kifejezést is alkalmazhatunk paraméterként. Természetesen ekkor a hívott gépi kódú rutint a sztringparaméter átvételére kell felkészíteni. Erről egy gyakorlati példa kíséretében a közeljövőben adok részletes tájékoztatást.

Frank Kampow *BASIC-gyakorlatok a Commodore 64-esen* című könyvében (Data Becker—Novotrade kiadás) is furcsa dolgokat találtam. Az USR(X) függvényről a 107. oldalon azt állítja, hogy „ SYS-hez hasonlóan működik, azzal a különbséggel, hogy lehetőségünk van a BASIC és a gépi nyelvű program közötti értékadásra". A 220. oldalon pedig — megerősítve, hogy nem véletlen elírásról van szó — a következő állítást olvashatjuk: „Az X értéke átadásra kerül a gépi rutinnak, amelyik a BASIC programnak egy másik értéket ad vissza. Ezzel lehetőségünk van a kétféle program közti adatátvitelre, ami a SYS-szel nem valósítható meg."

Óriási tévedés! Míg az USR-nek csak egy paramétere lehet, a SYS hívásakor átadható paraméterek számát csak a BASIC utasításszor mérete korlátozza. Az előbbinek a használatát bonyolítja még, hogy hívás előtt a gépi kódú rutin belépési pontjának a címét — POKE utasításokkal — be kell írni az USR vektorba. Ha egy programon belül több különböző felhasználói függvényre van szükségünk, minden hívás előtt az aktuálisnak a címét a vektorba kell tölteni. Az IBM PC BASIC-jében érdekes lehetőség van ennek a problémának a megoldására. Tíz függvény lehet egyidejűleg definiálva, ezekre USRn(x) alakban szabad hivatkozni, ahol n egy 0 és 9 közé eső szám. A függvények belépési pontjainak címei is egyszerűen meghatározhatók, a DEF USRn = <címkifejezés> szintaktikájú definiáló utasítással. Jól jönne valami hasonló a jelen sorozatban ismertetett géptípusokon is. A közeljövőben megvizsgáljuk ennek a lehetőségét.

Előfordul, hogy az USR függvénnyel dolgoznak a SYS utasítás helyett. Azokon a géptípusokon, melyeknek a BASIC-jéből a SYS hiányzik, ez szükségszerű, de a Commodore gépeken felesleges. Jó példa a helytelen alkalmazásra az 1986. júliusi számban megjelent C16 APPEND program, melynek ide vonatkozó részletét a *3. listán* láthatjuk. A két POKE és az értékadás helyett elég lenne egyetlen SYS 4104 utasítás. Az USR függvény felesleges alkalmazásáról árulkodik az, hogy az x változónak való értékadás a későbbiekben nincs kihasználva.

Az USR vektor gépi kódú rutinból való beállítására Németh Béla cikkében találunk példát (Hasznos programok az USR függvény alapján. Mikroszámítógép Magazin, 1987/7. szám). Előnye csak látszólagos, mert amit a vámon nyerünk, elveszítjük a réven: a beállító rutint SYS utasítással kell aktivizálni.

BARNA LÁSZLÓ

1. lista

```
100 poke 785,250 : poke 786,163
110 for i=1 to 8 : read x
120 print usr(x) : next
130 data 65536, -66000
140 data .000000000000000000036
150 data 200000000000000000000000000000
160 data 1.7e8, 1e38, 3.2e-37, -2e37
```

2. lista

```
 65536
-66000
 3.6e-21
 2e+28
 170000000
 1e+38
 3.2e-37
-2e+37
```

3. lista

```
  ...
8 poke1281,8:poke1282,16
9 x=usr(0):stop
  ...
```

Z80 programok haladóknak Spectrumra és Primóra

7. A manószerkesztő Primós változata

Az előző számban közölt manószerkesztő rutinoknak a Primóhoz szükséges változtatásait a mostani lista tartalmazza.

Vigyázzunk, a Primo beépített karakterkiviteli rutinja sokban különbözik a Spectrumétól! Ha a manószerkesztőt Primón is akarjuk futtatni, sok átalakításra van szükség. Két lehetőség közül választhatunk: vagy átírjuk a Spectrum (egyszerűsített) karakterkiviteli rutinját a karakterkészlettel együtt, vagy figyelembe véve a különbségeket, a programot írjuk át sok helyen. Mindenesetre megadom a listában a Spectrum karakterkiviteli rutinját, egyszerűsítve (EMIT: színkezelés, INVERSE, OVER nélkül). Ezzel remélem, működni fog a program. Az ehhez szükséges karakterkészlet ugyanolyan, mint a Spectrumé.

```
;-----------------------------------;
;      Srdef 2.1   I/O rutinok      ;
;        UHI-software BME           ;
;      modositasok Primohoz         ;
;-----------------------------------;
SCR     EQU   ...   ;kepernyo cime
              ;legyen oszthato 32-vel!
SCBUF   EQU   ...   ;szabad,$1800 hosz-
              ;szu teruletre mut.
INK           ;elhagyando
PAP           ;elhagyando
UDG     DEFS 2      ;UDG szimulaciohoz
CURADD  DEFS 2      ;cursor cime
CHARS   DEFW ...    ;a spectrum tipusu
            ;karakterkeszlet kezdocime
;-----------------------------------;
OPEN2   LD    HL,SCR        ;cursor-cim
        LD    (CURADD),HL   ;inicializ.
        RET                 ;   (=home)
;-----------------------------------;
ATBC    LD    HL,SCR        ;kepernyo cime
        LD    A,B           ;sor
        CP    24            ;ellenorzes
        RET   NC            ;sor)24, semmi
        LD    A,C           ;oszlop
        AND   $1F           ;0...31 lehet!
        LD    C,A           ;visszatolt
        ADD   HL,BC         ;pozicio beall
        LD    (CURADD),HL   ;uj cursor
        RET
;-----------------------------------;
EMIT    PUSH BC         ;karakterkivitel
        PUSH DE
        PUSH HL
        PUSH AF
        LD   DE,EEND    ;visszateresi
        PUSH DE         ;cim betoltes
        LD   HL,(CURADD)  ;cursorcim
        CP   " "        ;space?
        JR   NC,EMITCH  ;kiirhato chr
        CP   6          ;tabulator?
        JR   Z,ECOMMA   ;igen
        CP   8          ;backster?
        JR   Z,EBACK    ;igen
        CP   CR         ;carriage ret
        RET  NZ         ;nem: semmit
ECR     LD   L,0        ;oszlop
        INC  H          ;kov. sor
        LD   A,H        ;
        CP   SCREND/256 ;ellenorzes
        RET  C          ;nem lepte tul
        DEC  H          ;cursor fel
        CALL SCROLL     ;kep emelgetes
        POP  BC         ;eend cim torl
EEND    LD   (CURADD),HL  ;cur tarol
        POP  AF
        POP  HL
        POP  DE
        POP  BC
        RET
;-----------------------------------;
SCROLL  PUSH HL
        LD   HL,SCR+256 ;masodik sor
        LD   DE,SCR     ;elso sor
        LD   BC,$16FF   ;23sor hossza
        LDIR            ;sorok mozg.
        LD   H,D        ;utolso sor
        LD   L,E        ;   cime
        INC  DE         ;kov bajtja
        LD   (HL),0     ;az utolso sor
        LD   BC,$FF     ;   torlese
        LDIR            ;...
        POP  HL
        RET
;-----------------------------------;
ECOMMA  LD   A," "      ;space
        CALL EMITCH     ;kiirja
        LD   A,L        ;vizsg: elerte
        AND  $0F        ;mar a felsort
        RET  Z          ;igen
        JR   ECOMMA     ;folytatas
;-----------------------------------;
EBACK   LD   A,L        ;ellenorzes:
        OR   A          ;sor eleje?
        RET  Z          ;igen: semmi
        DEC  L          ;balra ler
        RET
```

```
;-----------------------------------;
EMITCH LD   BC,(CHARS);karakterkesz
       CP   #80       ;kod ell
       JR   C,EMIT1   ;irhato
       CP   #90       ;udg eleje
       RET  C         ;kisebb,nem ir
       SUB  #70
       LD   BC,(UDG)  ;udg kar-keszl
EMIT1  SUB  #20       ;0... kodtart
       EX   DE,HL     ;DE=cursorcim
       LD   H,0       ;HL=A
       LD   L,A       ;...
       ADD  HL,HL     ;HL=HL*8+BC
       ADD  HL,HL     ;...
       ADD  HL,HL     ;...
       ADD  HL,BC     ;...
       EX   DE,HL     ;DE=chr cime
       PUSH HL        ;HL=cursorcim
       LD   B,8       ;pixel-sorok
EMIT2  LD   A,(DE)    ;karakter rajz
       LD   (HL),A    ;kepre kerul
       LD   A,L       ;kov. pixelsor
       ADD  A,#20     ;...
       LD   L,A       ;...
       INC  DE        ;chr kov. bajt
       DJNZ EMIT2     ;cikluszaras
       POP  HL        ;eredeti curs
       INC  L         ;elorelep
       LD   A,L       ;ellenorzes
       AND  #1F       ;sorvege-e
       RET  NZ        ;meg nem
       JP   ECR       ;ha igen
;-----------------------------------;
PIXADD ...            ;a 4.reszben szereplo
                      ;pixadd rutin
;-----------------------------------;
CLS    PUSH BC
       PUSH DE
       PUSH HL
       LD   HL,SCR    ;kepcim
       LD   DE,SCR+1
       LD   (HL),0
       LD   BC,SCRLEN-1
       LDIR           ;0-val tolti
       POP  HL
       POP  DE
       POP  BC
       RET
;-----------------------------------;
IYDOWN PUSH BC        ;a HLDOWN-hoz
       LD   BC,32     ;hasonlo...
       ADD  IY,BC
       POP  BC
       RET
;-----------------------------------;
PAUSE0 CALL #001D ;keyboard handler
       JR   Z,PAUSE0  ;nem volt bill
       RET
;-----------------------------------;
WAIT   PUSH BC
       LD   BC,200    ;ugy kell be-
;allitani, hogy kb 320ms-ot varjon
WAIT1  CALL #001D     ;KYBHND
       RET  NZ        ;volt bill
       DEC  BC        ;varakozas cs
       LD   A,B
       OR   C
       JR   NZ,WAIT1  ;ciklusz aras
       RET
;-----------------------------------;
GET    CALL MSG       ;kiirja a
       DEFB "_",8,EOL ;      cursort
       CALL PAUSE0    ;b-nyomast var
       CALL PIP       ;csipog
       PUSH AF        ;bill-kod
       CALL MSG       ;leveszi a
       DEFB " ",8,EOL ;      cursort
       POP  AF        ;a megny bill
       RET
;-----------------------------------;
OPENIN PUSH BC
       LD   BC,#1500  ;a 23.sorra
       CALL ATBC      ; pozicional
       POP  BC
       RET
;-----------------------------------;
CLRIN  PUSH BC        ; torli az u-
       PUSH DE        ; tolso 2 sort
       PUSH HL        ;...
       LD   HL,SCR+176*32
       LD   DE,SCR+176*32+1
       LD   BC,16*32-1
       LD   (HL),0
       LDIR
       POP  HL
       POP  DE
       POP  BC
       RET
;-----------------------------------;
PIP    PUSH AF
       LD   A,7       ;primo-bell
       CALL #0015     ;DSPHND (ROM)
       POP  AF
       RET
;-----------------------------------;
KEYIN  ...            ;az eleje marad
KEYIN2 CP   CR        ;tehat 2 sor kima-
       RET  NZ        ; radt
       LD   A,"E"
       RET
;-----------------------------------;
; A kovetkezo rutinokbol ki kell
;hagyni a szinvezerlo karakterek ki-
;iratasat (INK,PAP,19) es a hozzajuk
;tartozo parametert (kovetkezo chr)!
;Ezenkivul a pozicionalasokat (pl:
;AT,x,y) at kell irni a kov. alakra:
       LD   BC,poz
       CALL ATBC
;ahol poz felso bajtja a sor,also az
;oszlopszam.
LISTSP ...
LISTPT ...
CLEAR  ...
;-----------------------------------;
```

Számítógépes grafika Pascalban

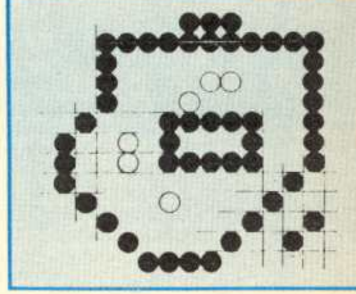
ZX–SPECTRUM

KIFESTÉS

Folytatjuk azt a cikksorozatot, amelyben a grafikai programozás alapjait mutatjuk be a Sinclair ZX-Spectrum Hisoft Pascal programnyelvén. Az eddigiekben a pont, egyenes szakasz, kör, körív és ellipszis rajzolására adtunk eljárásokat. Most bemutatjuk, hogy a BASIC POINT függvényének Pascal megfelelőjét hogyan lehet megszerkeszteni és mire lehet használni.

Lássuk először is magát a függvényt! A Point segítségével megállapíthatjuk, hogy az X és Y képernyő-koordinátákkal megadott pont tinta- vagy papírszínű-e. Az előzőekhez hasonlóan a Point függvényhez felhasználjuk a ROM megfelelő rutinját. Az *1. lista* elején találjuk meg a kérdéses függvényt. A FUNCTION fejében megadott X és Y paramétert a BC regiszterpárba töltjük, majd az eredeti rutinnal szemben az érvényes Y tartományt 0—191 pontra terjesztjük ki (90-es sor), hogy a vizsgálatot a képernyő egész területén elvégezhessük. A boolean típusú függvény értéke igaz, ha a pont tintaszínű, és hamis, ha papírszínű. A programsorokba kapcsos zárójelek között az utasítások Z80 assembly megfelelőjét is beírtuk. Ezek és a ROM listája segítségével a programrész működése könnyen követhető.

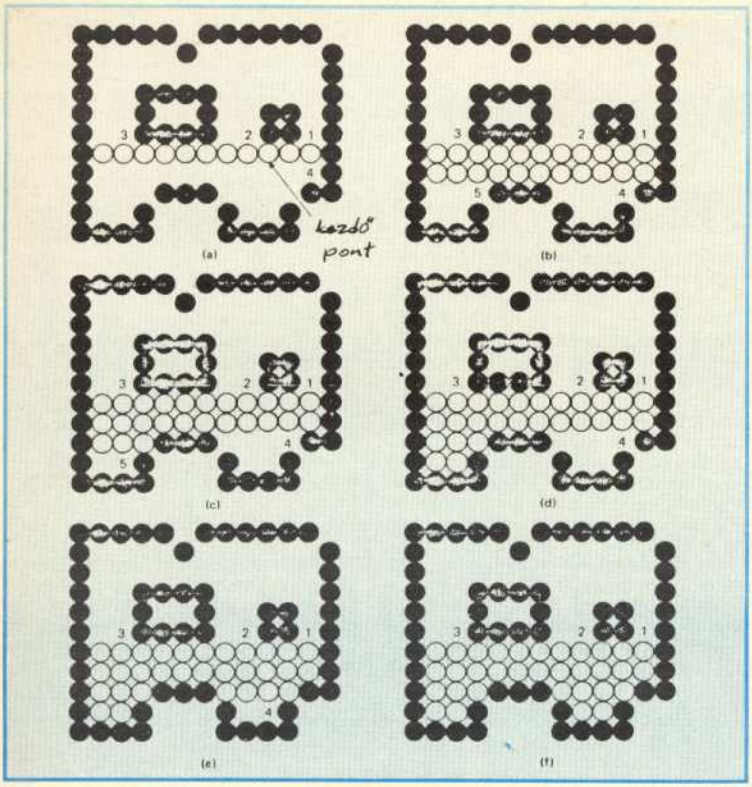
No, de mire is lehet ezt a függvényt használni? Természetesen mindarra, amire a BASIC megfelelőjét. Nekem személy szerint mégis úgy tűnik, hogy a Pascal-változat sokkal hasznosabb, mivel nagyságrendekkel gyorsabban működik. A BASIC POINT-ja annyira lassú, hogy ha olyan feladattal találkoztam is, amelyhez felhasználhattam volna, inkább gépi kódú szubrutinnal helyettesítettem. Erre, mint látni fogjuk, a Pascal esetében általában nincs szükség.



1. ábra

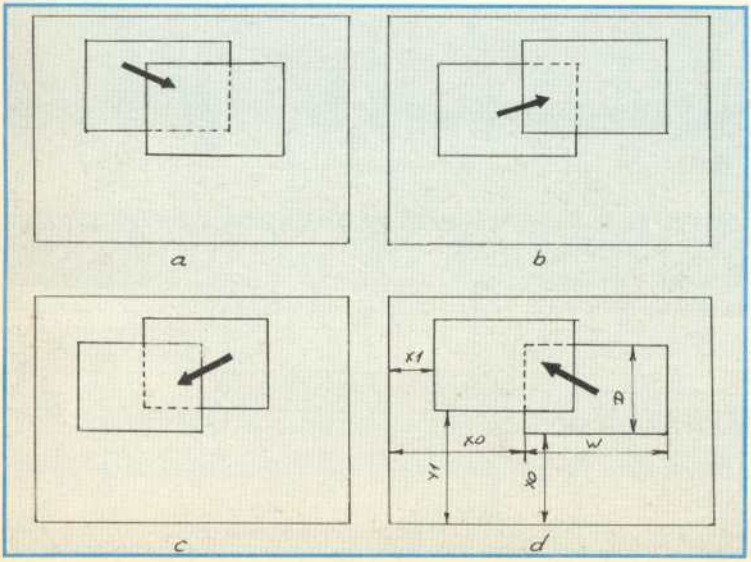
1. lista

```
10 (*L-)
20 (I:GETPOI File nev alatt kimentett
21 INCLUDE rutinok)
30
40 FUNCTION Point(X,Y:INTEGER):BOOLEAN;
50 BEGIN
60   INLINE(
70   #DD,#46,#02,  (LD B,(IX+2))
80   #DD,#4E,#04,  (LD C,(IX+4))
90   #3E,#BF,      (LD A,191)
100  #CD,#AC,#22,  (CALL #22AC)
110  #47,          (LD B,A)
120  #04,          (INC B)
130  #7E,          (LD A,(HL))
140  #07,          (RLCA)
150  #10,#FD,      (DJNZ -2)
160  #E6,#01,      (AND 1)
170  #DD,#77,#02   (LD (IX+2),A));
180   IF Y=1 THEN Point:=TRUE
181   ELSE Point:=FALSE
190 END; (Point)
200
210 PROCEDURE Fill(X,Y:INTEGER);
220 (Rekurziv algoritmus negyiranyu
221 lepteteseel)
230 BEGIN
240   IF Point(X,Y)<>TRUE THEN
250   BEGIN
260     PLOT(X,Y);
270     Fill(X,Y-1);
280     Fill(X,Y+1);
290     Fill(X-1,Y);
300     Fill(X+1,Y)
310   END
320 END; (Fill)
330
340 PROCEDURE Fill8(X,Y:INTEGER);
350 (Rekurziv algoritmus nyolciranyu
351 lepteteseel)
360 BEGIN
370   IF Point(X,Y)<>TRUE THEN
380   BEGIN
390     PLOT(X,Y);
400     Fill8(X,Y-1);
410     Fill8(X,Y+1);
420     Fill8(X-1,Y-1);
430     Fill8(X-1,Y);
440     Fill8(X-1,Y+1);
450     Fill8(X+1,Y-1);
460     Fill8(X+1,Y);
470     Fill8(X+1,Y+1)
480   END
490 END; (Fill8)
500
510 PROCEDURE FILL(X,Y:INTEGER);
520 (Smith nemrekurziv algoritmusa)
530 CONST StackDepth=20;
540 TYPE  Coordinate=(x,y);
550 VAR   Stack:ARRAY [1..StackDepth,x..y]
551            OF INTEGER;
560        i,j  :INTEGER;
570
580   PROCEDURE Run(X,Y:INTEGER);
590   VAR i,k:INTEGER;
600       B  :BOOLEAN;
610   BEGIN
620     i:=X;
630     j:=j-1;
640     B:=TRUE;
650     WHILE (NOT Point(i,Y)) AND (i>=0) DO
660     BEGIN
670       IF Y<191 THEN
680       IF NOT Point(i,Y+1) THEN
690       BEGIN
700         IF B THEN
710         BEGIN
720           j:=j+1;
730           Stack[j,x]:=i;
740           Stack[j,y]:=Y+1;
750           B:=FALSE
760         END
770       END ELSE
780         B:=TRUE;
790       PLOT(i,Y);
800       i:=i-1
810     END;
820     k:=i+1;
830     i:=X;
840     B:=TRUE;
850     WHILE i>=k DO
860     BEGIN
870       IF Y>0 THEN
880       IF NOT Point(i,Y-1) THEN
890       BEGIN
900         IF B THEN
910         BEGIN
920           j:=j+1;
930           Stack[j,x]:=i;
940           Stack[j,y]:=Y-1;
950           B:=FALSE
960         END
970       END ELSE
980         B:=TRUE;
990       i:=i-1;
1000    END
1010  END; (Run)
1020
1030 BEGIN
1040   i:=X;
1050   j:=1;
1060   WHILE (NOT Point(i,Y)) AND (i<256) DO
1070     i:=i+1;
1080   Run(i-1,Y);
1090     WHILE j>0 DO
1100       Run(Stack[j,x],Stack[j,y])
1110 END; (FILL)
1120
1130 PROCEDURE CopyField(X0,Y0,X1,Y1,D,W:
1131 INTEGER);
1140 (Teglalapny terulet atmasolasa)
1150 VAR X,Y:INTEGER;
1160 BEGIN
1170   FOR X:=0 TO W DO
1180   FOR Y:=0 TO D DO
1190     IF Point(X0+X,Y0+Y) THEN
1191     PLOT(X1+X,Y1+Y)
1200 END; (CopyField)
1210
1220 PROCEDURE TurnField(X0,Y0,X1,Y1,D,W:
1221 INTEGER);
1230 (Teglalapny terulet elforgatasa 90 fokkal)
1240 VAR X,Y:INTEGER;
1250 BEGIN
1260   FOR X:=0 TO W DO
1270   FOR Y:=0 TO D DO
1280     IF Point(X0+X,Y0+Y) THEN
1281     PLOT(X1-Y,Y1+X)
1290 END; (TurnFiled)
1300
1310 PROCEDURE MirrorY(X0,Y0,X1,Y1,W,D:
1311 INTEGER);
1320 (Teglalapny terulet tukrozese
1321 az Y tengelyre)
1330 VAR X,Y:INTEGER;
1340 BEGIN
1350   FOR X:=0 TO W DO
1360   FOR Y:=0 TO D DO
1370     IF Point(X0+X,Y0+Y) THEN
1371     PLOT(X1-X,Y1+Y)
1380 END; (MirrorY)
1390
1400 PROCEDURE MirrorX(X0,Y0,X1,Y1,W,D:
1401 INTEGER);
1410 (Teglalapny terulet tukrozese
1411 az X tengelyre)
1420 VAR X,Y:INTEGER;
1430 BEGIN
1440   FOR X:=0 TO W DO
1450   FOR Y:=0 TO D DO
1460     IF Point(X0+X,Y0+Y) THEN
1461     PLOT(X1+X,Y1-Y)
1470 END; (MirrorX)
1480
1490 PROCEDURE Triangle(X1,Y1,X2,Y2,X3,Y3,V:
1491 INTEGER);
1500 (Haromszog rajzolasa)
1510 BEGIN
1520   FatLine(X1,Y1,X2,Y2,V);
1530   FatLine(X2,Y2,X3,Y3,V);
1540   FatLine(X3,Y3,X1,Y1,V)
1550 END; (Triangle)
1560
1570 PROCEDURE Rectangle(X1,Y1,X2,Y2,
1571 X3,Y3,X4,Y4:INTEGER);
1580 (Negyszog rajzolasa)
1590 BEGIN
1600   PixelLine(X1,Y1,X2,Y2);
1610   PixelLine(X2,Y2,X3,Y3);
1620   PixelLine(X3,Y3,X4,Y4);
1630   PixelLine(X4,Y4,X1,Y1)
1640 END; (Rectangle)
1650
L
```



2. ábra. A számok azoknak a pontoknak a helyét jelölik, amelyeket a program a veremtárban tárol

3. ábra



Készítsünk olyan eljárást, amellyel egy tetszőleges alakú területet be tudunk „festeni". Ilyen utasítás a ROM-ban nincs, de egyes BASIC-kiterjesztésekben, például a Beta BASIC-ben „Fill" kulcsszó alatt megtalálható. Úgy kell használni, hogy rámutatunk egy körülhatárolt képernyőterület tetszőleges pontjára, és meghívjuk az eljárást.

Az 1. listán több festőeljárást is találunk. Sorrendben az elsőt Fill néven definiáltuk. Ez igen egyszerű felépítésű. Az eljárás megvizsgálja, hogy az eljárásfej X, Y paramétereivel megadott pont papírszínű-e. Ha igen, akkor tintaszínnel kifesti, majd megvizsgálja a pont bal oldali, jobb oldali, alsó és felső szomszédját, és megismétli a műveletet. Ez egészen egyszerű: a pont felrajzolása után az eljárás (procedure) megfelelően megváltoztatott paraméterekkel négyszer egymás után önmagát hívja.

Az ilyen algoritmust, amely saját magát hívó részeket tartalmaz, rekurzív algoritmusnak hívják, és általában azért alkalmazzák, mert — a mi példánkban is látható — elegáns, tömör alakban fogalmazható meg. Mindenesetre nagyon kell vigyázni, mert könnyen előfordulhat, hogy kifutunk a számítógép memóriájából. Hogy lehet ez? Az eljárás egyszeri hívása nemcsak négy további hívással jár, hanem annyival, amennyi szükséges a terület teljes befestéséhez. A programlistából ugyan nem látható, de tudnunk kell, hogy minden egyes eljáráshívásnál a Pascal veremtárába (stack) el kell helyezni a visszatérési címet. Az így lefoglalt hely ugyan később felszabadul, de csak akkor, ha az eljárás már befejeződött, és a vezérlést a hívó programrész kapja vissza. A helyzet hasonló ahhoz, amit a borbélynál tapasztalunk, ha belenézünk az előttünk lévő tükörbe. Ekkor nemcsak a hátunk mögött lévő falat látjuk, hanem, mivel ott is tükör van, tükrök, borbélyok, falak és vendégek sokaságát, holott csak két tükröt használunk.

Ha valaki a fentiek igazáról meg akar győződni, próbáljon egyre nagyobb zárt alakzatokat rajzolni a képernyőre és fesse ki a Fill eljárással. Szomorúan tapasztalhatja, hogy a program futása már viszonylag kis méretű sikidomok esetén is megszakad memóriahiány miatt. Természetesen írhatunk más elven dolgozó festőeljárást, melynél ez a hátrány nem jelentkezik, de először vizsgáljuk meg ennek egy másik hiányosságát.

Ha különböző alakú területek kifestésével próbálkozunk, hamarosan tapasztaljuk, hogy az algoritmus nem működik jól. Nem hajlandó például az eljárás bizonyos esetekben a hegyes szögek csúcsa melletti pontokat kifesteni. Ha egy összefüggő terület valahol nagyon összeszűkül, az eljárás hibásan úgy érzi, hogy ott kontúr van, és nem fest tovább. Ha elemezzük az ilyen helyeket *(1. ábra)*, könnyen rájövünk az okára: a terület nem a vízszintes, illetve függőleges irányban elhelyezkedő szomszédos pontokkal folytatódik, hanem átlós irány-

ban kellene továbblépni, amire történetesen ez az eljárás nem képes.

Az *1. lista* Fill8 nevű eljárása már nem négy, hanem nyolc irányban vizsgálja a szomszédos pontokat, tehát a hibát könnyen kijavítottuk és lám, nem is sokkal bonyolultabb a program! Korai örülni, próbáljuk ki először az eljárást. Megdöbbenünk: a ferde kontúrvonalak mentén a festék kibújik a területünkről, az eljárás nem működik. Nincs mit tenni: higabb festéknél jobban kell vigyázni a festőnek, nehogy lecsorogjon. Ha a területet nem a PixelLine eljárással rajzolt egyenesekkel, hanem vastagabb vonalakkal határoljuk, a „réseket" betömjük, minden a várakozásnak megfelelően alakul.

Most térjünk rá a rekurziót helyettesítő, nem memóriafaló festőeljárás szerkesztésének problémájára. Sajnos fel kell adnunk a Fill eljárás eleganciáját és rövidségét. Az egyik megoldás szerint, melynek megvalósítása az *1. lista* FILL eljárásában, működése pedig a *2. ábrán* látható, egy Stack azonosítójú INTEGER tömbben tároljuk a terület nevezetes pontjainak X, Y koordinátáit. A Stack (veremtár) név nem véletlen, az adattömböt valóban veremtárként fogjuk használni. A j INTEGER változó lesz a veremmutató, vagyis az a szám, amely mindig a veremtár első üres helyére mutat. A Stackbe mindig a j-edik elemet írjuk be vagy a j—1-edik elemet olvassuk ki, írás után j-t eggyel növeljük, olvasáskor pedig értékét eggyel csökkentjük. A veremtár által lekötött memória a főprogramba való visszatéréskor felszabadul, mivel a Stack a FILL eljárásra nézve lokális változó.

Az eljárás először itt is megvizsgálja a terület belsejére mutató pontot, hogy papírszínű-e, majd addig lép jobbra, amíg tintaszínűt talál, vagyis a terület határára ér. Ekkor meghívja a FILL eljáráson belül definiált RUN eljárást, amely a következőket teszi. A paraméterként megadott X, Y koordinátájú pontból kiindulva addig lép balra, amíg a terület határához nem ér, vagyis tintaszínű pontot nem talál. Eközben befesti az Y magasságban húzódó pontsort, és megjegyzi az Y+1 pontsor azon pontjainak a koordinátáit, amelyek a területhatártól közvetlenül balra helyezkednek el. Ez a „megjegyzés" abból áll, hogy a koordinátákat beírja a Stackbe.

Ezután hasonlóan balról jobbra végigvizsgálja az Y magasságban húzódó pontsor alatti pontokat is, a területhatártól balra lévő pontok koordinátáit itt is a Stackbe rakja. A FILL eljárás egy WHILE ciklussal addig hívja a RUN eljárást a veremtár tetején lévő pont koordinátáival, amíg a veremtár ki nem ürül. Ekkor a festés befejeződik.

A veremtár mélységét a StackDepth állandóval 20-ra állítottuk be, ami az esetek többségében több mint elegendő. A demo programban bemutatott példában a veremtárat csak 6 hely mélységben vesszük igénybe. Ha azonban a veremtár mérete valakinek mégis kicsinek bizonyulna, írjon be bátran nagyobb értéket. Az eljárást úgy alakítottuk ki, hogy a képernyő szélét is területhatárnak tekinti. Így gyakorlatilag „bolondállónak" tekinthető: hacsak a Stackből ki nem futunk, nem lehet a rutint rosszul hívni.

A Point függvény használatára másik példa néhány további eljárás, mellyel a képernyőn a rasztermintát kihasználó különböző manipulációkat végzünk. Ezek közül a legegyszerűbb a CopyField nevű eljárás, amellyel a képernyő egy téglalap alakú területét lehet egy másik részre átmásolni *(3. ábra)*. Az eredeti terület bal alsó sarkának koordinátái XO, YO, a terület magassága D, szélessége W, az új terület bal alsó sarkának koordinátái pedig X1, Y1.

Az eljárás roppant egyszerű: egy X és egy Y szerint futó kettős ciklusban megvizsgáljuk, hogy minden egyes pont papírvagy tintaszínű-e. Ha tintaszínű, az új terület megfelelő pontjára a PLOT eljárás segítségével pontot rajzolunk. Az eljárás ebben a formájában rámásolja a másolandó képet arra a képre, amely a kijelölt új területen éppen található, az eredetit pedig változatlanul hagyja. Természetesen lehet olyan követelmény, hogy az eredetit törölje, az új területre pedig csak a másolandó kép kerüljön, az eredetileg ott lévőt a program rajzolja felül, „takarja le". Ezeknek a programváriánsoknak az elkészítését az olvasóra bízzuk. Segítségképpen emlékeztetünk arra, hogy a SPOUT(CHR(21)); SPOUT(CHR))1; utasításokkal a BASIC OVER 1 attributuma érhető el a Pascalban.

Más a probléma, ha a másolandó és a célterület egymást részben átfedi. Ekkor nem biztos, hogy az eljárásunk helyesen működik, mert lehet, hogy már átmásolt területrészeket fog tovább másolni. A lehetséges eseteket a *3. ábra* szemlélteti. Vegyük észre, hogy a CopyField csak a c) és a d) esetben dolgozik a várakozásnak megfelelően. Természetesen az összes esetre lehet készíteni jól működő eljárást, sőt olyan eljárás is szerkeszthető, amely az összes esetben jól működik, csak a másolás sorrendjét kell megfelelően megadni: ezt is az olvasóra bízzuk.

A TurnField eljárás a képernyő egy téglalap alakú területét az óramutató járásával ellenkező irányban, 90 fokkal elforgatja. A MirrorX és MirrorY eljárások viszont az x, illetve y tengelyre tükröznek. Ezekbe a műveletekbe mindig egy x, illetve y tengely irányú eltolás is beleértendő.

Eljárásgyűjteményünket még két egyszerű taggal egészítettük ki: a Triangle és a Rectangle eljárással, amelyekkel háromszög, illetve négyszög rajzolható. Ezeknek a Point függvényhez semmi közük. Csak azért kerültek ide, mert a demo programban *(2. lista)* célszerű felhasználni őket. A demonstrációs program magáért beszél. az előzőek alapján úgy érezzük, minden részlete érthető.

DR. KABOLDY PÉTER

```
10 (*L-*)
20 PROGRAM FILL;
30 VAR I,J,X,Y:INTEGER;
40     b       :BOOLEAN;
50 (*L+*)
60 (*F 1:LINE  *)
70
80 (*L-*)
90 (*F 1:GETPOI*)
100
110 BEGIN
120   PAGE;
130   PixelLine(100,100,100,130);
140   PixelLine(100,130,110,130);
150   PixelLine(110,130,110,110);
160   PixelLine(110,110,120,110);
170   PixelLine(120,110,120,150);
180   PixelLine(120,150,130,150);
190   PixelLine(130,150,130,100);
200   PixelLine(130,100,100,100);
210   Fill(105,105);
220   Rectangle(10,10,10,40,40,40,40,10);
230   Rectangle(20,20,20,30,30,30,30,20);
240   Fill(15,15);
250   Triangle(160,80,200,120,190,100,2);
260   Fill8(175,93);
270   CopyField(10,10,200,140,30,30);
280   TurnField(160,80,50,130,40,40);
290   MirrorY(10,130,100,130,40,40);
300   MirrorX(10,130,10,120,40,40);
310   WRITE(CHR(22),CHR(0),CHR(0));
320   WRITELN('Nyomjon meg egy gombot!');
330   TurnField(0,183,248,10,8,180);
340   TurnField(240,10,239,10,180,8);
350   MirrorX(0,183,0,183,180,8);
360   REPEAT UNTIL INCH()CHR(0);
370   PAGE;
380   FOR I:=0 TO 4 DO
390   FOR J:=0 TO 4 DO
400   BEGIN
410     X:=I*50+5;
420     Y:=J*38+5;
430     Rectangle(X,Y+15,X+20,Y,X+40,Y+15,X+20,Y+30)
440   END;
450   FILL(110,80);
460   REPEAT UNTIL INCH()CHR(0)
470 END.
```

2. lista

Függvények és utasítások II.

A függvényszerű nyelvek

Az első részben áttekintettük a függvények és eljárások definiálását és végrehajtását. Most a függvények gyakorlati hasznosítását boncolgatjuk tovább.

Természetesen a felhasználó által definiált függvényeket nemcsak magából a főprogramból lehet hívni, hanem a bonyolultabb függvények definíciójába beépíthetjük az egyszerűbbek hívását is. Felmerül a kérdés: nem lehetne-e a programot pusztán függvényekkel megoldani? Úgy képzelhetjük el a dolgot, hogy a felhasználó az adott elemi függvényekből kiindulva egyre nagyobbakat hoz létre, amíg egy olyan összetett függvényt nem kap, amely magának a programnak a feladatát látja el. A program futtatása ezek után: meghívjuk a programot leíró függvényt, és argumentumaiként megadjuk a bemenő adatokat. A végrehajtás után megkapjuk az eredményt.

Könnyű elképzelni, hogy egy ilyen nyelvet valóban meg is lehet csinálni. A teoretikusokat azonban az izgatta, hogy vajon leírható-e egy ilyen nyelvvel is az összes olyan program, amely az utasításokat tartalmazó nyelvekkel leírható.

A válasz igen is — nem is. Ahhoz, hogy ezt megértsük, tisztáznunk kell a függvények kiszámításának pontos menetét.

Függvénykiszámítási módok

A matematikában egymásba ágyazott függvényeket úgy értékelünk ki, hogy először a belső függvények értékét számítjuk ki és rendeljük hozzá a paraméterekhez, majd ennek segítségével határozzuk meg a külső függényt. Ezt a kiszámítási módot *érték szerinti hívásnak* (call by value) nevezzük. Például a sin(d*3.1415/180) azaz fokokban megadott szöget először átszámoljuk radiánba a 3.1415/180-as konstanssal való szorzással, majd alkalmazzuk rá a szinusz függvényt.

Nem ez azonban az összetett függvényhívás egyetlen lehetséges kiszámítási sorrendje. Egy másik módszer az ún. *szükség szerinti hívás* (call by need). Ebben az esetben rögtön a külső függvény kiszámításához kezdünk hozzá, és az argumentum értékét csak akkor keressük, amikor először szükség van rá.

További utak is vannak — az ún. *név szerinti* kiszámítási módról később lesz szó.

Első pillanatban arra gondolhatunk, hogy a két kiszámítási mód mindig azonos eredményre vezet. Ez azonban csak addig igaz, amíg teljes (totális) függvényekről van szó, amelyek minden argumentumuk minden lehetséges értékére értelmes eredményt adnak. Amint áttérünk a parciális (nem mindenütt értelmezett) függvényekre, egészen más eredményt kapunk. A függvényt akkor nevezzük parciálisnak, ha bizonyos argumentumértéknél nem ad eredményt (hiba vagy végtelen ciklus keletkezik). Ilyen például az osztás, amelynél az osztó nem lehet nulla.

A parciális függvények körében egy összetett függvény kiszámítása egészen más eredményre vezethet az érték szerinti és a szükség szerinti kiszámítási módnál. Az első esetben az összetett függvény értéke biztosan definiálatlan lesz, ha a külső függvény egyik argumentumában egy olyan belső függvény áll, amely az adott helyen definiálatlan. A szükség szerinti hívás esetén lehet, hogy erre az argumentumra nincs is szükség, mert a többi már meghatározza a függvény értékét.

Tipikus példa erre a logikai függvények kiszámítása. Itt a szükség szerinti hívást *rövidzár-kiszámításnak* nevezik és így szól: az AND függvény értéke mindjárt hamis lesz, ha első argumentuma hamis; az OR függvény értéke mindjárt igaz, ha az első argumentuma igaz.

Ha például azt akarjuk leírni, hogy az A változó osztója B-nek, akkor ezt a következőképpen tehetjük:

B osztója A=(<>0) AND ((B MOD A)=0).

Ez persze csak akkor írható le így, ha az AND függvényt rövidzár módon számoljuk ki, különben nulla esetén azon nyomban hibajelzést kapunk a nullával való osztás miatt. Ha az AND függvény érték szerinti hívású, akkor csak egy elágazó utasítással válik ez az érték kiszámíthatóvá. Valahogy így:

B osztója A=IF A=0 THEN FALSE ELSE (B MOD A)=0.

Programkonstrukció

Hosszasan vizsgálható a kérdés, hogy ha vannak bizonyos alaputasításaink vagy alapfüggvényeink, akkor milyen konstrukciós eszközökre van szükségünk ahhoz, hogy ezekből bármely programot felépíthessünk. Végül az derül ki, hogy három ilyen konstrukciós eszköz van, melyek a következők:

Szekvencia. Szükség van arra, hogy előírhassuk: bizonyos lépéseket egymás után kell végrehajtani úgy, hogy minden lépés felhasználja az előző eredményét.

Elágazás. Szükség van arra, hogy valamilyen feltételtől függően vagy az egyik vagy a másik tevékenységsorozatot hajtsuk végre (és csak azt hajtsuk végre, a másikat ne).

Ciklus. Szükség van arra, hogy egy tevékenységet addig ismételjünk — minden lépésben kissé változó adatokkal —, amíg valamilyen feltétel be nem következik.

Az utasításokból felépülő nyelveknél a szekvenciát azzal biztosítjuk, hogy az utasításokat, amíg nincs elágazás, a gép egymás után hajtja végre. Az elágazást régebben a nyelvek a feltételes ugrással valósították meg, az újabb ugrásmentes nyelvek az IF... THEN... ELSE... konstrukcióval (ha a feltétel teljesül, a THEN utáni utasítássorozat kerül végrehajtásra, ha nem, az ELSE utáni). Gyakran adott a nyelvekben a CASE utasítás is, ami nem kétfelé, hanem többfelé ágazik, erre azonban csak a kényelem és hatékonyság kedvéért van szükség. A ciklust a régebbi nyelvek visszaugrással valósították meg. A modern nyelvekben — így az ugrásmentes nyelvekben is — számos ciklusutasítás van. A WHILE ciklus addig ismétel, míg a fejében levő feltétel teljesül: a FOR ciklusnak egy változója van, és a ciklus úgy ismétlődik, hogy ez a változó felvesz bizonyos értékeket a kezdőértéktől a végértékig; a LOOP ciklus mindaddig ismétlődik, amíg egy benne elhelyezett utasítás (BREAK vagy EXIT) feltétele nem teljesül.

Persze ezeknek az utasításoknak a konkrét formája nyelvenként más és más. Kialakult azonban, hogy körülbelül milyen utasítás is az, amit FOR ciklusnak, WHILE ciklusnak, LOOP-nak, CASE-nek, IF-nek hívnak.

A függvényszerű nyelvekben a szekvenciát az határozza meg, hogy az érték szerint hívott függvényeknél az argumentumhoz rendelt belső függvényt előbb számoljuk ki,

mint a külsőt. Elágazásra az ad lehetőséget, hogy bizonyos függvényeknél nem érték szerint, hanem szükség szerint számoljuk ki az argumentumok értékét.

Tipikus elágazó függvény a COND, melynek tetszőleges számú argumentumpárja van. A pár első tagja egy logikai kifejezés. Ha ez igaz, kiszámoljuk a pár második tagját; ez lesz a függvény eredménye. Ha az első tag nem volt igaz, a pár második tagját nem számoljuk ki, hanem a következő párt vizsgáljuk.

A függvényszerű nyelvekben a ciklust rekurzióval valósíthatjuk meg.

Rekurzív függvények

Rekurziónak nevezzük, ha egy függvény vagy eljárás meghívja önmagát. Kölcsönösen rekurzívnak nevezünk két vagy több függvényt, ha meghívják egymást; így indirekten a függvény önmagát is meghívhatja. Kimutatták, hogy ami kölcsönös rekurzióval leírható, az leírható egyszerű rekurzióval is, csak egy sokkal bonyolultabb függvény kell hozzá. Mivel azonban a kölcsönös rekurzió megvalósítása sem költségesebb, használata viszont lényegesen kényelmesebb, azok a programozási nyelvek, amelyek megengedik a rekurziót, megengedik a kölcsönöset is. Például listának nevezünk egy olyan adatszerkezetet, amely elemek felsorolásából áll, és egy elem vagy egy szó, vagy egy allista. A **Head,** azaz fej nevű függvény visszaadja a lista első elemét. Az **Atom** nevű függvény pedig igaz értéket ad, ha az adott elem egy szó, és hamisat, ha az adott elem lista. A **First,** azaz első nevű függvény visszaadja a listában szereplő első szót, ha a lista első eleme szó: ha az első elem lista és benne az első elem szó, akkor azt és így tovább, függetlenül attól, hogy milyen mélyen beágyazva találja meg az első szót a listában:

First(x:lista) = COND(
(**Atom**(**Head**(x)), **Head**(x)),
(TRUE, **First**(**Head**(x)))

Annak, hogy egy függvény meghívja önmagát, természetesen csak úgy van értelme, ha valami feltételtől függően és az eredetitől eltérő argumentumokkal hívja meg önmagát, hiszen különben végtelen ciklusban tenné. Ezt azonban nem szokás vizsgálni, hiszen az első látásra legtökéletesebb függvénynél is előfordulhat (akár szándékosan, akár hibából), hogy bizonyos argumentumoknál mégis végtelen hívási sorozat keletkezik. Hasonló ez ahhoz, ahogy egy WHILE ciklusnak is lehet olyan feltétele, ami bizonyos körülmények között örökké teljesül, és ezért a program végtelen ciklusba jut. A nagy különbség az, hogy mivel a rekurzív függvényhívás mindig felhasznál egy kis darabot a memóriából, a végtelen rekurzió egyszer mindig véget ér hibajelzéssel, mivel elfogy a memória.

Ebből az is látható, hogy hiába indulunk ki totális függvényekből, az eredmény nem lesz feltétlenül totális függvény.

A LISP és társai

Az első függvényszerű nyelvet, a LISP-et 1958-ban definiálta J. McCharty. 1960-ra implementálták a LISP 1.5 nyelvet, amely nemcsak függvényszerű elemeket tartalmazott, de részként tartalmazta az 58-as ún. „tiszta" LISP-et, és formai tekintetben nagyon hasonlított rá. Azóta is újra meg újra felmerül a vágy, hogy vissza kellene térni az eredeti LISP gondolatokhoz és egy tisztán függvényszerű nyelvet kellene létrehozni és használni. Született is egy sor ilyen nyelv, amelyek természetesen már további, újabb gondolatokat is tartalmaztak.

Az egyik legújabb ilyen nyelv a Miranda, amely elődeire, a SASL, KCR, HOPE és ML nyelvekre támaszkodik. Nem mondhatnám, hogy valamelyik is ezek közül különösebben elterjedt volna a gyakorlatban, de elméletileg érdekesek, és időnként a belőlük származó gondolatok hatással vannak a konvencionális utasításszerkezetű nyelvekre is.

A LISP nyelv viszont tagadhatatlanul igen nagy karriert futott be az elmúlt 25 év során. A mesterséges intelligencia alapvető nyelve volt hosszú ideig, és Amerikában még ma is az. Előnyei közé tartozik, hogy névvel ellátott szimbólumok listáján dolgozik, és ezek a listák — valamint a listákból összetett listák és más ehhez hasonló adatstruktúrák — a program futása közben dinamikusan keletkeznek és halnak meg. Az interpreter maga gazdálkodik a memóriával, a felhasználónak nem kell vele törődnie. Másik előnye, hogy nemcsak az adatok vannak listában ábrázolva, hanem a függvények is, és futás közben olyan adatokat hozhatunk létre, amelyek azután függvénydefinícióvá válnak és részt vesznek a program további végrehajtásában. Végül, de nem utolsósorban lényeges tulajdonsága, hogy a nyelv pontos működése viszonylag nagyon röviden és matematikai szabatossággal meg van adva. A megadás tartalmaz néhány kiinduló függvényt és azokat a szabályokat, hogy miként kell ezekből újabbakat létrehozni és hogyan kell az így létrehozott függvények meghívását értelmezni. *(Folytatjuk.)*

FARKAS ERNŐ

Pályázati felhívás

Az NJSZT, a szekszárdi Garay János Gimnázium és a Mikroszámítógép Magazin Szerkesztősége 1988-ra is programozási pályázatot hirdet általános iskolás, középiskolás diákok és elsőéves egyetemi hallgatók részére az alábbi kategóriákban:

1. Új, önálló játékprogram készítése HT—1080Z, Sinclair Spectrum, Commodore 16 és 64 és Videoton TV—Computer gépre.
2. Valamely tantárgyhoz kapcsolódó, a tanítási órát segítő, illetve a tanulók önálló tanulását támogató oktatóprogram készítése a fent említett gépekhez.
3. A zsűri az idén először külön is elbírálja a pályázatokhoz benyújtott programleírásokat (dokumentáció), a legjobb pályaműveket a verseny szervezői különdíjjal jutalmazzák.

A pályázat benyújtási határideje: **1988. január 31.**

A programot mágnesszalag-kazettán (a kazettára többször felvéve), vagy mágneslemezen (floppy) rövid leírás kíséretében (mit tud, hogyan működik a játék, hányadik osztály mely tantárgyához, melyik anyagrészhez kapcsolódik stb.), jeligével ellátva (külön zárt borítékban a név, lakcím vagy iskola) kérjük beküldeni a Mikroszámítógép Magazin Szerkesztősége (1371 Budapest, Pf.: 433) címére.

A szerkesztőségnek joga van a pályázaton részt vett programok közlésére, amiért a szokásos honoráriumot fizeti. A döntő 10—10 résztvevőjét az NJSZT tagjaiból, a Garay Gimnázium tanáraiból és a szerkesztőség munkatársaiból álló előzsűri választja ki. A zsűrinek joga van — ha nem érkezik be elegendő pályázat, vagy a pályaművek színvonala nem éri el a kívánt szintet — a döntő résztvevőinek számát csökkenteni.

A döntő — melyen az előzsűri által kiválasztott programok versenyeznek — Szekszárdon, a Garay János Gimnáziumban rendezendő Garay-napok alkalmából, 1988. március 14-én és 15-én lesz.

A döntőbe jutott tanulókat a Garay János Gimnázium vendégül látja.

Az 1. és 2. kategóriában az első három helyezett programot díjazzuk, és mindkét kategóriában kiadjuk a közönség díját is.

NJSZT
Dömölki Bálint
elnök
Garay János Ált. Gimn.
Zentai András
igazgató
Mikroszámítógép Magazin
Szerkesztősége
Kovács Győző
a szerkesztőbizottság
vezetője

Szoftver

Cikksorozatunkban először a gépi kódú programozást segítő három legfontosabb eszközzel, az *EDITOR*, az *ASSEMBLER* és a *MONITOR* programmal kapcsolatos alapvető ismereteket mutatjuk be. Aki gépi nyelven akar programozni, annak ezekre a programokra feltétlenül szüksége van.

A MONITOR

A MONITOR-okról általában

A MONITOR egy gépi kódban megírt, általában 1—4 kbájt nagyságú „kisegítőprogram". Segítségével tudja a felhasználó a saját gépi szintű programjának működését ellenőrizni és az előforduló hibákat javítani. Ezt a folyamatot a program „belövésének" nevezzük. A számítástechnikai zsargonban a debug (ejtsd: dibag = poloskairtás) kifejezés is eléggé elterjedt. A program belövése során általában a következő funkciókra van szükség:

— kijelölt memóriatartalom vagy adott regisztertartalom megjelenítése(kiíratása) és esetleges módosítása,
— program indítása,
— töréspont elhelyezése a programban,
— lépésenkénti programvégrehajtás,
— program rögzítése háttértárolón és visszatöltése.

Tartalom megjelenítése és esetleges módosítása

A felhasználó programjának tesztelése, belövése során gyakran van szükség a tár egyes rekeszeinek vagy egy egész tártartománynak kilistázására a képernyőre. Ezzel egyrészt a program egyes utasításai, másrészt a program által használt változók ellenőrizhetők és a kívánt értékre módosíthatók. Az erre szolgáló parancs segítségével tudunk például kisebb rutinokat beírni a tárba, vagy a már meglévő proram hibás utasításait átírni. A kilistázott és beírt adatok mind hexadecimális értékek. Ebben a számrendszerben tizenhat különböző számjegy van: 0123456789ABCDEF. Következésképpen 0AH (H a hexadecimális számrendszert jelzi) értéke a tizes számrendszerben 10, 0BH = 11, . . . 0FH = 15, ezután következik 10H, ami a decimális számrendszerben 16.

A „belövés" alatt álló programban lévő hibák gyors felderitését hatékonyan segítheti, ha a program egyes jellemző pontjain a processzor regisztereinek — amelyeknek tartalmán az utasítások különböző műveleteket végeznek — aktuális értékét ki tudjuk iratni a képernyőre. Ezzel gyorsan megtalálható az a pont (utasítás, programlépés), ameddig a program jól működik és amely után a hibát keresnünk kell. Szintén alapvető elvárás egy MONITOR-tól e regiszterek tartalmának módosíthatósága. Erre például akkor lehet szükség, ha csak egy programrészt akarunk vizsgálni, és ennek tesztelése előtt a regisztereket egy meghatározott tartalommal fel kell tölteni.

Program indítása

A MONITOR-nak ez a funkciója teszi lehetővé, hogy a felhasználó egyrészt a rendszerben levő többi programot (EDITOR, ASSEMBLER), másrészt a saját maga által írt és futtatni kívánt programokat megindíthassa. Az indítási (futtatási) parancs mögött mindig szerepel egy hexadecimális szám, amely a program kezdőcime.

A programnak a futás végén mindig vissza kell térnie a MONITOR-ba.

Töréspont elhelyezése a programban

A program belövése során igen nagy segítséget nyújt, ha a programot egy, még az indítás előtt kijelölt címen is meg lehet állítani. Ekkor ugyanis — mint már emlitettük — a regiszterek és a programban használt változók tartalmainak vizsgálatával könnyen eldönthető, hogy a program eddig a pontig helyesen vagy hibásan működött. A teljes programot több, logikailag jól elhatárolt részre bontva a hibásan működő programrész rendkivül gyorsan meghatározható.

A program leállítása úgy történhet, hogy a kérdéses címen levő utasítást egy ugróutasításra cseréljük, mely a vezérlést — ha a program futása során az adott címet elérte — a MONITOR-nak adja át.

Természetesen ilyen leállítási feltételt — azaz töréspontot, angolul breakpoint — csak RAM-ban (Random Access Memory = véletlen elérésű tár, azaz irható és olvasható is) elhelyezett programba lehet beiktatni. ROM-ban (Read Only Memory = csak olvasható tár) vagy EPROM-ban (Electrically Programable Read Only Memory = elektronikusan programozható, csak olvasható tár) lévő programoknál a töréspont így természetesen nem alkalmazható. A RAM-beli program továbbindítása előtt a MONITOR a töréspont címén lévő eredeti utasítást visszaírja a helyére, és erről a címről folytatódik a program futása.

Lépésenkénti programvégrehajtás

Ha a töréspontok elhelyezésével sikerült meghatározni azt a programrészt, amelyben a program hibásan működik, akkor e részen belül a hibás utasítás vagy utasítássorozat megtalálásának módja a lépésenkénti programvégrehajtás. Ebben az üzemmódban ugyanis a felhasználói programnak (a belövés alatt álló programnak) mindig csak egy utasítása hajtódik végre, s így lehetőség van a regiszterek vagy a tár vizsgálatára, vagyis a végrehajtott utasítás helyes működésének ellenőrzésére.

Program rögzítése háttértárolón és visszatöltése

A programok fejlesztése mindig a tár RAM részében történik. A RAM-ok a tápfeszültség kikapcsolása után elveszítik a bennük lévő információt. Mivel egy program alapos belövése hosszabb időt is igénybe vehet, ezért szükség van a program tárolására valamilyen módon. Erre szolgálnak az ún. háttértárak, amelyek aránylag nagy tömegű adat befogadására képesek. Ezek általában mágneses tárolók, például magnókazetta vagy lemez. Régebben a lyukszalagot is kiterjedten alkalmazták.

TICK JÓZSEF—DR. KÓNYA LÁSZLÓ

CHIP

(Bhattacharya, P.K.: Commodore 64: Garbage Collection. Chip, 1986. dec. 188—192. old. Fordította: Tóth Ferenc, ALUTERV—FKI)

Szemétgyűjtés

A sztringmemória-terület gyors nagytakarítása

Aki olyan programokat ír, amelyekben sok sztringművelet van, előbb vagy utóbb tapasztalni fogja, hogy a gép munka közben percekig állni látszik. Ennek az az oka, hogy értékadásnál az új sztringet a gép nem a réginek a helyére teszi, hanem új helyre. Eközben egyszer csak betelik a szabad memóriaterület. A látszólagos semmittevés alatt a gép átvizsgálja az operatív memória tartalmát, és abból kitakarítja az időközben már feleslegessé vált sztringeket.

Az *1. lista* szerinti program a sztringmemória-terület gyors nagytakarítására alkalmas rutin. A C64 ROM-jában levő nagyon lassú rendszerrutinnál sokkal gyorsabban dolgozik.

A program tulajdonképpen kompromisszum: összesen és legfeljebb 2048 sztringet tud kezelni, amelyek a $33/$34 . . . .$37/$38 címeken levő sztringterületen helyezkednek el. Ez esetben a sztringek hossza tetszőleges lehet, de természetesen csak max. 255 karakterből állhat egy-egy sztring. Ha feltesszük, hogy egy-egy szövegsor hossza átlagosan 80 karakter, akkor az összes sztringek teljes hossza $80 \times 2048 = 163840$ karakter lehetne, ami sokkal nagyobb a C64 memóriájánál.

Ha azonban egy sztring csupán egyetlen karakterből állna, ami a szövegfeldolgozás esetén szinte sohasem fordul elő, akkor ez meghaladná a program képességeit. Ekkor a program hibajelzéssel leáll, és visszalép a BASIC szintre.

A programnak két startcíme van. Az 1. startcím, az 51 200 ($C800) úgy viselkedik, mint a PRINT FRE(0) funkció. Elvégzi a sztringterület nagytakarítását, kiszámítja és kiírja a szabad memóriaterületet.

Azért, hogy elkerüljük a C64 saját rutinjának a működését, beiktattuk a 2. startot, az 51 203 címen ($C803). Ha ezt hívjuk, akkor kiszámítja, hogy a szabad memóriaterület kisebb-e 510 bájtnál. Ez két maximális hosszúságú sztringnek felel meg.

A C64 eredeti rutinja minden sztringértékadás után szabad memóriahelyet keres a sztring számára. Ha a szabad terület kisebb a szükségesnél, a rutin ugyan elvégzi a nagytakarítást, de ez nagyon lassan zajlik le.

Azért, hogy ezt megelőzzük, a programot hívjuk a 2. startnál. Ez megállapítja a szabad sztringterület méretét, és már akkor elvégzi a gyors nagytakarítást, amikor a ROM-rutin még nem lépne működésbe. Azt érjük így el, hogy a rendszerrutin soha nem fog lassan nagytakarítani. Ehhez az szükséges, hogy a sztringértékadásokat tartalmazó programokba minden értékadás (pl. A$="...) előtt hívjuk a 2. startot (SYS 51203). Mivel ekkor rendszerint nem zajlik le a nagytakarítás, ez szinte alig vesz igénybe futási időt.

A nagytakarítás után a program még egyszer ellenőrzi a szabad memóriaterület méretét. Ha ez még ekkor is 510 bájtnál kisebb lenne, figyelmeztető jelzés jelenik meg a képernyőn. Ekkor a felhasználónak kell valamilyen intézkedéssel szabad sztringterületet felszabadítania, mert különben a rendszerrutin rövidesen működésbe lép, és mivel az sem tud új területet felszabadítani, a program „out of memory" (memória-túlcsordulás) hibajelzéssel leáll.

Az 1. start és a 2. start jelű programrészek célja eltérő. Mindkettő a „nagytakarítás" programrészt hívja. Mivel ennek a programrésznek is szüksége van a memória 00. oldalán (zero page) néhány bájtnyi memóriahelyre, ezek tartalmát előbb átmenetileg ki kell menteni. Mivel ugyane memóriahelyeket a megszakító (interrupt) rutinok kiterjesztésére is használhatná a rendszer, a program a megszakítást kikapcsolja. Egyébként a program a RAM-ban foglal helyet, közvetlenül a ROM-terület alatt a $F000 . . .$FFFF című részen. Amikor aktívvá válik, a ROM-ot időlegesen kikapcsolja. Amiatt is szükséges a megszakítási rutin kikapcsolása, mert ez is főként a ROM-ban foglal helyet.

A program először is megvizsgálja, hogy van-e egyáltalán szemét a memóriában. Ehhez előbb kiszámítja azon sztringek összes hosszát, amelyek a sztringmemória területén ($33/$34 . . .$37/$38) vannak. Ha nincs szemét, a kiszámított sztringhosszak összege egyenlő a BASIC-RAM vége ($37/$38) és a sztringterület kezdete ($33/$34) közötti különbséggel. Egyéb tennivaló tehát nincs. Ha van szemét, akkor a sztringterületre mutató összes sztringtömb és sztringváltozó címét áthelyezi az operatív memória (RAM) $F000 . . $FFFF területére. Az áthelyezés során a címeket növekvő sorrendbe rendezi, és ezzel olyan leíró (deszkriptor) címlistát készít, amit aztán felhasznál a sztringek áthelyezésénél, eltolásánál.

A sztringek eltolása a leírólista alapján alulról ($F000) felfelé halad. Ez minden sztringet érint, amelyiknek címe a sztringterületen a BASIC-RAM terület végétől lefelé található. Vagyis az első az a sztring lesz, amelyik közvetlenül a BASIC—RAM vége után következik.

Ezután sztringről sztringre haladva kiszámítja a szükséges címeket, majd ellenőrzi ezeket a tényleges címek felhasználásával. Ha ekkor különbség adódik, a sztringet eltolja a szükséges címre, és az új címet rendeli hozzá a sztringleíróhoz. Ha valamelyik következő sztringleíró is ugyanerre a sztringre mutat, akkor már csak a sztring új címét írja be a leíróba.

A szemétgyűjtő program működésének bemutatására készült a demonstrációs program *(2.lista)*. Ennek négy része van, és jól mutatja be a nagytakarító program képességeit és korlátait. Az első részben (10-40-es sorok) kiszámítja és a képernyőre írja a szabad memóriahelyek számát. Ekkor még nincs szemét a memóriában. Ezután a program szemétsztringeket hoz létre, és ekkor a rendező rész is aktívvá válik. Jelezni fogja, hogy a sztringek száma túllépte a megengedett határt, és visszalép a BASIC-be.

A második részben (50-80-as sorok) a sztringek számát a megengedett határon belül tartja úgy, hogy a program működhessen.

```
10 REM "SZTMEMTI B-BET"
20 REM BASIC BETOLTO A SZTRINGMEMORIA
30 REM TISZTITO PROGRAMHOZ
40 REM CHIP 86. DEC. P188-192
50 REM (C) P. K. BHATTACHARYA
60 REM FORD. & ATDOLG. TOTH F. 87.03.10
100 DATA 4C,62,C8,20,46,C8,B0,4F,3A3
101 DATA 20,58,C8,20,7F,C8,4E,56,34B
102 DATA CB,B0,66,20,46,C8,B0,61,420
103 DATA A5,D3,8D,57,CB,A5,D6,8D,52F
104 DATA 58,CB,A9,20,A0,CB,20,1E,395
105 DATA AB,AE,53,CB,AD,54,CB,20,463
106 DATA CD,BD,A9,32,A0,CB,20,1E,40E
107 DATA AB,AC,57,CB,AE,58,CB,18,462
108 DATA 20,0A,E5,4C,79,C8,A5,33,374
109 DATA 38,E5,31,8D,53,CB,A5,34,3D2
110 DATA E5,32,8D,54,CB,C9,02,60,3EE
111 DATA A5,01,8D,5F,CB,A9,37,85,3C2
112 DATA 01,60,20,58,C8,20,7F,C8,308
113 DATA 4E,56,CB,B0,0C,20,46,C8,359
114 DATA AE,53,CB,AD,54,CB,20,CD,485
115 DATA BD,AD,5F,CB,85,01,60,78,3F2
116 DATA A2,07,B5,F7,9D,57,CB,CA,4DE
117 DATA 10,F8,A9,00,8D,56,CB,85,3E4
118 DATA F9,85,FA,A5,2D,85,F7,A5,56B
119 DATA 2E,85,F8,A5,F8,C5,30,90,4CD
120 DATA 08,D0,26,A5,F7,C5,2F,B0,43E
121 DATA 20,A0,00,B1,F7,30,11,C8,371
122 DATA B1,F7,10,0C,C8,20,68,CA,3DE
123 DATA 4E,56,CB,B0,03,20,A7,CA,3B3
124 DATA 20,BD,CA,4C,9B,C8,20,C9,43F
125 DATA CA,A5,F8,C5,32,90,08,D0,4C6
126 DATA 33,A5,F7,C5,31,B0,2D,A0,442
127 DATA 00,B1,F7,30,E9,C8,B1,F7,531
128 DATA 10,E4,20,88,CA,A5,F8,CD,4D0
129 DATA 54,CB,D0,07,A5,F7,CD,53,4B2
130 DATA CB,F0,D6,20,66,CA,4E,56,485
131 DATA CB,B0,03,20,A7,CA,20,B7,3E6
132 DATA CA,4C,E5,C8,A5,37,38,E5,4BC
133 DATA F9,AA,A5,38,E5,FA,C5,34,558
134 DATA D0,07,E4,33,D0,03,4C,07,314
135 DATA CB,A5,2D,85,F7,A5,2E,85,471
136 DATA F8,20,DC,CA,A8,91,F9,C8,5B8
137 DATA D0,FB,E6,FA,D0,F7,A9,F0,70B
138 DATA 85,FA,D0,03,20,BD,CA,A5,49E
139 DATA F8,C5,30,90,08,D0,2C,A5,426
140 DATA F7,C5,2F,B0,26,A0,00,B1,412
141 DATA F7,30,E9,C8,B1,F7,10,E4,574
142 DATA 20,B4,CA,20,66,CA,4E,56,392
143 DATA CB,B0,07,86,FD,85,FE,20,4A8
144 DATA 14,CA,20,BA,CA,4C,37,C9,3CE
145 DATA 20,C9,CA,A5,F8,C5,32,90,4D7
146 DATA 08,D0,37,A5,F7,C5,31,B0,451
147 DATA 31,A0,00,B1,F7,30,E9,C8,45A
148 DATA B1,F7,10,E4,20,88,CA,A5,4B3
149 DATA F8,CD,54,CB,D0,07,A5,F7,557
150 DATA CD,53,CB,F0,D6,20,66,CA,501
151 DATA 4E,56,CB,B0,07,86,FD,85,42E
152 DATA FE,20,14,CA,20,B7,CA,4C,3E9
153 DATA 87,C9,A5,37,85,33,A5,38,3C1
154 DATA 85,34,20,DC,CA,85,FE,20,422
155 DATA E5,CA,AA,F0,54,88,B1,FB,5D1
156 DATA 8D,55,CB,C8,B1,FB,AA,C8,593
157 DATA B1,FB,C5,FE,D0,04,E4,FD,624
158 DATA F0,2F,85,F8,86,F7,48,A5,506
159 DATA 33,38,ED,55,CB,85,33,B0,3E0
160 DATA 02,C6,34,68,C5,34,D0,04,331
161 DATA E4,33,F0,0D,AC,55,CB,88,468
162 DATA B1,F7,91,33,88,C0,FF,D0,583
163 DATA F7,A5,F7,85,FD,A5,F8,85,637
164 DATA FE,A0,01,A5,33,91,FB,C8,4CB
165 DATA A5,34,91,FB,20,F9,CA,D0,518
166 DATA A6,4C,07,CB,A5,F9,48,A5,44F
167 DATA FA,48,A5,F9,38,E9,02,85,488
168 DATA F9,B0,08,C6,FA,A5,FA,C9,5D9
169 DATA F0,90,1F,20,E5,CA,B1,FB,51A
170 DATA AA,C8,B1,FB,C5,FE,90,06,577
171 DATA D0,10,E4,FD,B0,0C,A5,FB,51D
172 DATA 91,F9,C8,A5,FC,91,F9,4C,5C9
173 DATA 1A,CA,A0,02,A5,F7,91,F9,4AC
174 DATA C8,A5,F8,91,F9,68,85,FA,5D6
175 DATA 68,85,F9,20,F9,CA,D0,05,49E
176 DATA 68,68,4C,13,CB,60,A0,00,2FA
177 DATA B1,F7,F0,16,8D,55,CB,C8,523
178 DATA B1,F7,AA,C8,B1,F7,C5,2E,5B5
179 DATA 90,08,D0,0B,E4,2D,F0,02,376
180 DATA B0,05,A9,01,8D,56,CB,60,36D
181 DATA C8,B1,F7,18,65,F7,8D,53,4C4
182 DATA CB,C8,B1,F7,65,F8,8D,54,579
183 DATA CB,C8,B1,F7,AA,20,BA,CA,589
184 DATA 20,B4,CA,CA,D0,FA,60,AD,53F
185 DATA 55,CB,18,65,F9,85,F9,90,4A4
186 DATA 02,E6,FA,60,A9,02,2C,A9,3C2
187 DATA 03,2C,A9,05,2C,A9,07,18,1D1
188 DATA 65,F7,85,F7,90,02,E6,F8,548
189 DATA 60,A0,03,B1,F7,48,88,B1,42C
190 DATA F7,18,65,F7,85,F7,68,65,4B4
191 DATA F8,85,F8,60,A9,F0,85,FA,5ED
192 DATA A9,00,85,F9,60,A9,35,85,3EA
193 DATA 01,A0,00,B1,F9,85,FB,C8,493
194 DATA B1,F9,85,FC,A2,37,86,01,48B
195 DATA 60,A5,F9,18,69,02,85,F9,3FF
196 DATA 90,02,E6,FA,A5,FA,60,A2,513
197 DATA 07,BD,57,CB,95,F7,CA,10,44C
198 DATA F8,58,60,A9,40,A0,CB,20,424
199 DATA 1E,AB,20,82,CA,4C,07,CB,353
200 DATA 0D,11,53,49,45,20,48,41,1A8
201 DATA 42,45,4E,20,4E,4F,43,48,21D
202 DATA 20,00,20,42,59,54,45,53,1C7
203 DATA 20,46,52,45,49,20,20,00,186
204 DATA 0D,5A,55,20,56,49,45,4C,20C
205 DATA 45,20,53,54,52,49,4E,47,23C
206 DATA 53,0D,00,00,00,00,00,00,060
290 :
```

Néhány másodperc alatt végrehajtja a szemét nagytakaritását (alig vesszük észre), ami a C64 BASIC-jének körülbelül hat percig tartana.

A harmadik részben (100-160-as sorok) a nagytakaritó program 2. starttal való hívásának hasznosságát mutatja be. A bal felső sarokban kis négyzet villogása jelzi, amint minden ciklusban megvizsgálja, hogy mennyi szabad memóriahely van még. Ha a négyzet villogása rövid időre megáll, az mutatja, hogy ezalatt zajlik le a

```
300 SI=54272:FOR I=0 TO 20:POKE SI+I,0:
301 NEXT:POKE SI+24,15:REM SID
310 PRINT "■"
320 X=35:XX=33:FOR I=0 TO 106:
321 FOR J=0 TO 7
330 : READ Z$:GOSUB 500:IF F=9 THEN 700
340 : POKE 51200+8*I+J,D:DM=DM+D:NEXT J
350 :READ Z$:GOSUB 500:DM=DM-D
360 :IF DM<>0 THEN A$="OSSZEG":X=10:
361 :IF DM<>0 THEN XX=17:ZZ=500:GOTO700
370 PRINT "■";I+100:GOSUB 600:NEXT I:
371 RESTORE
390 :
400 AD=51200:HI=INT(AD/256):LO=AD-256*HI
410 INPUT "GEPI KODU PROGRAM NEVE";A$
420 OPEN 1,8,1,A$:
421 PRINT#1,CHR$(LO);CHR$(HI);
430 FOR I=0 TO 106:FOR J=0 TO 7
440 : READ Z$:GOSUB 500
450 PRINT#1,CHR$(D);:NEXT J
460 READ Z$:NEXT I
470 CLOSE 1:GOTO 800
490 :
500 D=0:FOR S=1 TO LEN (Z$):
501 H=ASC(MID$(Z$,S))-48:H=H+7*(H>9)
510 :IF H>16 OR H<0 THEN F=9:A$="IRAS":
511 :IF H>16 OR H<0 THEN X=15:ZZ=500:
512 :IF H>16 OR H<0 THEN RETURN
520 D=D*16+H:NEXT S:RETURN
590 :
600 POKESI+1,X:POKESI+6,247:POKESI+4,XX
601 FOR Z=0 TO ZZ:NEXT Z:POKE SI+4,XX-1
602 RETURN
690 :
700 GOSUB 600:PRINT "■■  "A$"HIBA A";
701 PRINT 100+I;". SORBAN!"
800 POKE SI+4,0:POKE SI+24,0
```

1. lista

```
5 REM "DEMO SZTMEMTI"
6 REM DEMO PROGRAM A SZTRINGMEMORIA
7 REM TISZTITO MUKODESENEK BEMUTATASARA
10 DIM A$(2048):PRINT "■ 'SZMT' DEMO"
11 REM "■"=<SHIFT>+<CLR/HOME>
20 FOR I=0 TO 2048:A$(I)=CHR$(65):NEXT:
21 PRINT "■ 1. START":REM "■"=<CRSR>
25 PRINT "■ MEG NINCS SZEMET!
30 SYS51200:A$(0)="BHATTACHARYA":
31 PRINT "■ SZEMET SZTRINGEK KESZITESE"
40 SYS51200:
41 PRINT "■ SZTRINGEK SZAMA TUL NAGY!"
45 :
50 CLR:DIM A$(2047)
60 FOR I=0 TO 2047:A$(I)=CHR$(65):NEXT:
61 PRINT "■ 1. START - SZEMET NELKUL!"
70 SYS51200:A$(0)="BHATTACHARYA":
71 PRINT "■ SZEMET SZTRINGEK KESZITESE"
80 PRINT:SYS51200:
81 PRINT"■ SZTRINGEK SZAMA NEM TUL SOK"
90 :
100 PRINT"■ HOSSZU SZTRINGEK KITOLTIK":
101 PRINT"A MEMORIAT EGESZEN A HATARIG"
110 CLR:DIM A$(799):A$=CHR$(65)
120 PRINT"■ 2. START. SZEMET SZTRINGEK":
121 PRINT "        UJBOLI LETREHOZASA"
130 FOR I=1 TO 799:A$(I)=A$+A$(I-1)
140 IF LEN(A$(I))>92 THEN A$(I)=""
150 PRINT"■■ ■":SYS51203:PRINT"■ ":NEXT
160 PRINT "■■■■ 1. START":SYS51200:
161 PRINT
190 :
200 PRINT SPC(12)"BHAT";
210 SYS51203:FOR I=0 TO 700
220 NEXT:SYS51203:PRINT "TACHARYA";
```

2. lista

nagytakarítás. Végül többször kapunk jelzést arra, hogy a memória megtelőben van.

Figyelmet érdemel, hogy maga a demonstrációs program is helyet foglal el a memóriában. Ez érzékelhetővé is tehető. Ha a demo programot lerövidítjük, például elhagyjuk a REM-eket, akkor nem jelenik meg a figyelmeztetés. Ha pedig meghosszabbítjuk a demo programot, például több REM sort írunk bele, akkor előfordulhat, hogy a C64 saját rutinja aktiválódik. Hosszabb várakozási idő után a program felvilágosít minket arról, hogy a gép „out of memory", vagyis túlcsordult.

A demo program negyedik része (200-220-as sorok) bizonyítja, hogy a figyelmeztető jelzés után a kurzor ugyanott jelenik meg újra, ahol előzőleg látható volt, tehát a figyelmeztetés után minden további nélkül folytathatjuk a munkát.

Ha a gépi kódú részt valamelyik monitorprogrammal adjuk be, akkor a DATA sorokban lévő hexadecimális számokat közvetlenül billentyűzhetjük, és nem kell beadni a sorok végén levő háromjegyű hexa számokat. Ezeket ugyanis csak a BASIC-betöltő használja fel a DATA sorok helyességének ellenőrzésére.

A nagytakarító program DATA listájának végén van a BASIC-betöltő rész. Ez a RUN után összehasonlítja a sorok első nyolc számának az összegét a kilencedik számmal, és ha nincs különbség, betölti azokat a megfelelő memóriahelyekre. Ezt a folyamatot a képernyőn is figyelemmel kísérhetjük: megjelenik az éppen ellenőrzött DATA sor száma, valamint hangjelzést is ad a gép. Ha nincs hiba a DATA sorokban, akkor megkérdezi, hogy milyen néven tárolja a gépi kódú programot, majd felveszi a mágneslemezre. Ha mágnesszalagon akarjuk tárolni a programot, akkor a 420-460-as sorokban a perifériakódot meg kell változtatni.

A program fontosabb adatai

Helye a memóriában:	$C800 . . .$CB60 (51200 . . 52064)
Munkaterülete:	$F000 . . .$FFFF (61440 . . .65535)
1. start címe:	$C800 (51200)
2. start címe:	$C803 (51203)

A sztringek száma a sztringmemória-területen max. 2048.

Miért nem stupid a MUPID? avagy: Milyen az intelligens videotex-dekóder? IV.

Sorozatunkban eddig áttekintettük a MUPID belső felépítését, főbb üzemmódjait, és a vtx-alkalmazások közül a fontosabb információbeviteli lehetőségeket. Ebben a részben a MUPID néhány lényegesebb videotexes rendszerprogramját, valamint jellemzőbb mikroszámítógépes alkalmazását mutatjuk be.

Videotex segédprogramok

Az M—DISK-hez mellékelt rendszerlemezre a MUPID cég a már említett szerkesztőprogramokon kívül egyéb, a videotexezést segítő programokat is elhelyezett. Ezek jó része az osztrák (részben az NSZK) vtx-központból is lehívható. Most csak a lényegesebbeket emeljük ki. A programok a lemezről menüből vagy *fájlnév#; a vtx-központból szintén menüből vagy közvetlen oldalszám megadásával (*oldalszám#) hívhatók le.

A **btxtodisk** program vtx-oldalak lemezre másolását támogatja, és a művelet általában problémamentes. Nem így a teleprogramok lemezre mentése! Egyrészt sok szerző, szoftverház védi portékáját valamilyen módon a másolás ellen, másrészt a roszszabb minőségű vtx-központ—vtx-terminál összeköttetésekre a megbízhatóság érdekében külön TSW-betöltést (TSW: a Telesoftware = teleprogram rövidítése) segítő program hívható le, amely nem mindig működik együtt a lemezre másoló programmal. A helyzetet az is bonyolítja, hogy ha a — legtöbbször BASIC nyelven megírt — TSW speciális utasításokat (SEND, FETCH stb. lásd később) tartalmaz, ezeket a fordítóprogram nem érti, és ezért további be/kiviteli rutinokat kell a gépbe tölteni. A MUPID „lelkivilágát" jobban ismerve, a problémák némi bűvészkedéssel természetesen megoldhatók.

A teleprogramoknál tartva: ezek betöltési ideje — ha sérülés miatt nem kell a központnak adatblokkot ismételnie — rövidebb programoknál 45—90 másodperc, de például a digitális képet megjelenítő vagy a NAPLPS megjelenítést értelmező programok esetében 5-6 perc. Az M—DISK-ről való lehívásnál ezek az időtartamok negyedére vagy tizenhatodára csökkennek.

A **line-dump** (vonali betöltés) és a **single step display** (lépésenkénti megjelenítés) segédprogramok lehetőséget nyújtanak a MUPID által vett és kiküldött karaktereknek (tehát a vezérlőkaraktereknek is) a megjelenítésére ASCII vagy hexadecimális kódban. A borítón látható Európa-térképen alfageometrikus oldalt leíró adatok karakterenkénti érvényre jutása figyelhető meg, miközben a képernyő alsó sorában az utoljára vett 8 karakter és hexadecimális kódja (kék háttérszínnel) jelenik meg.

```
Diskette  DRUCKPR        581a       MUPID
4. PRINT PROGRAMS


1 Printer-independent    Text-Print

EPSON-FX80, Mannesmann Tally MT85:
2 Text-Hardcopy (king-size)
3 Hardcopy (Alpha-mosaic, C2-Graphics
   HC standard )
4 Text-Print with special characters
   (Dip-Switch SW 1/4 "OFF")

Mannesmann Tally MT80+:
5 Hardcopy (Alpha-mosaic, C2-graphics
   HC standard )
Mannesmann Tally MT160:
6 Text-Hardcopy (king-size)
7 Hardcopy (Alpha-mosaic, C2-graphic,
   HC standard )

9 Explanation
0 Directory
```

1. ábra

Az **edit links** menüszervező program támogatja a szerkesztőt valamely lemezoldalon lévő vtx-oldalak tetszés szerinti egymáshoz láncolásának és menüből való lehívhatóságának kialakításában. Az így létrehozott alkalmazói (felhasználói) felület egy házon belüli (in-house) rendszeréhez hasonló, vagyis a lemezoldal a lehíváskor nemcsak nevével jelentkezik be, hanem mindjárt a rajta lévő fájlok menüválasztási lehetőségét is kínálja.

Hasonló a **demo** program használata, mellyel a lemezen lévő, mintegy 50-60 oldal egymás utáni automatikus megjelenésének sorrendje és időtartama előre programozható (áruházakban, boltok kirakataiban, várócsarnokokban való demonstráció céljaira).

A **framedump** (memóriából történő oldalbevitel) nevű program az információszolgáltatókat támogatja a lemezen előkészített vtx-oldalaik rendszerbe töltésében. A program futása során először a vtx-központ szerkesztőrendszerét kell meghívni, majd a szerkesztő jelszó megadása után választani kell a központ által felkínált lehetőségek (oldal törlése, új oldalak bevitele, meglévő oldal tartalmának vagy paramétereinek megváltoztatása) közül. Oldalparamétereken olyan jellemzőket kell érteni, mint az oldal azonosítószáma, melynek megadásával az oldal közvetlenül lehívható; ha aloldal, akkor melyik (ezt az angol ábécé kisbetűivel kell jelölni); van-e „zár" rendelve az oldalhoz, azaz korlátozott-e az oldalhoz való hozzáférés és milyen mértékben; az oldal ára (0,1—99,9 schilling) és végül a típusa, azaz információs vagy válaszoldal. Ezután kell megadni, hogy az M—DISK melyik olvasójában helyezkedik el az illető oldal és milyen néven, majd elkezdődik az oldal vtx-központba töltése, ami ugyan jóval egyszerűbb és gyorsabb a vonali szerkesztésnél, de több oldal bevitelénél a paraméterek megadása nagyon időigényes.

Ennek automatikussá tétele érdekében fejlesztették ki a külön lemezen található **auto-framedump** nevű programot. Ezzel

mindezek a műveletek helyben előre elvégezhetők, és a több tíz oldalra is kiterjeszthető betöltés sikerességéről a MUPID és a központ közötti párbeszéd alapján nyomtatott lista is készül.

Végül, de nem utolsósorban a már említett rendszerlemezen találhatók a különböző **nyomtatóprogramok**, melyek angol nyelvű menüje az *1. ábrán* látható.

Az eddigiek során — mint azt sorozatunk elején jeleztük — a MUPID-ot elsősorban a vtx-alkalmazások szemszögéből ismertettük. De most lássuk, milyen is

A MUPID mint mikroszámítógép

A hagyományos mikrogépfunkciók természetesen elemei ennek a gépnek is, jóllehet mindez (BASIC, monitor üzemmód, assembly programozás stb.) nem ROM-ban van, hanem külön kell RAM-ba tölteni. A gép kettős természetéből fakadóan igazi csemegeként kínálkozik, hogy egyrészt a MUPID mint mikroszámítógép például BASIC-ből erőforrásként meghívhatja a vtx-központot, másrészt a MUPID-ot mint vtx-dekódert támogatja a számtalan vtx-orientált alkalmazói program. Nézzük sorra az ismertebb lehetőségeket!

A MUPID **BASIC-ben** a rendszerlemezen található BPES (BASIC Program Entwicklungs System = BASIC fejlesztő rendszer) segítségével programozható, mely elsősorban az információszolgáltatónak nyújt támogatást a vtx-rendszerbe tölthető teleprogramok írásánál, de természetesen bárki használhatja helyi programírásra és a programok lemezre vagy kazettára mentésére is.

A BPES főbb sajátosságai:

— a programírást menü támogatja,

— az egyes BPES üzemmódok arra jellemző háttérszínnel is jól megkülönböztethetők (lásd a borítón).

```
1.BASIC program-fejlesztörendszer

1 Integer-BASIC
2 Lebegőpontos BASIC

3 BASIC újraindítás
4 Program újraindítása inicializálással
5 Program újraindítása inicializálás
  nélkül
6 Listázó program

7 BASIC segédprogramok


  9 Magyarázat
  0 Tartalomjegyzék
```

```
BASIC segédprogramok

1 Renumber
2 Párhuzamos betöltés és futtatás
3 Mozgató modulok
4 Hardcopy alprogram
5 Hardcopy Modul
6 Másolás elleni védés
7 CMOSIO Modul
8 80 karakteres modul


  9 Magyarázat
  0 Tartalomjegyzék
```

3. ábra

Speciális MUPID-BASIC utasítások:

Vtx-utasítások

TELANFANG	a vtx-központ automatikus hívása
TELENDE	a vtx-központtal való kapcsolat automatikus bontása
FETCH	a lehívott vtx-oldal sorainak egy string-tömb elemeihez történő rendelése
SEND	string küldése a vtx-központba (pl.közvetlen oldalszám)
CREATE	vtx-oldalak központba történő betöltése vagy lemezfájlok előállítása
CHAIN	programból újabb program behívása a vtx-központból

Grafikus utasítások

DRAW	grafikus elemek definiálását és megrajzolását megelőző utasítás
LIN	vonal rajzolása
BOG	körív rajzolása
ATL	vonalattributum beállítása
KRE	kör rajzolása
ATF	mintázat-attributum beállítása
TEX	grafikus írás
FD	szín definiálása

2. ábra

BASIC változatok:

— I(nteger)-BASIC: csak egész számú változókat enged meg, a vtx-specifikus alkalmazásoknál ez elegendő és gyorsabb műveletvégzést tesz lehetővé,

— G(leitkomma)-BASIC: lebegőpontos számábrázolás, ahol a számolási pontosság 7 és 61(!) digit között szabadon megadható.

Programfordítás: értelmezőprogram segítségével. A BASIC-ben írt szöveg a végleges fordítás előtt egy átmeneti tárban közbenső kódok formájában (3 programbájt = 4 karakter) tárolódik. Minden áttöltés (lemezre, vtx-központba stb.) ebben a TSW formátumban történik.

Maximális programhossz: 32 kbájt, mely egy speciális rendszerparanccsal 64 kbájtra növelhető.

Teszt üzemmódok: menüválasztásnál lépésenkénti üzemmód vagy nyomkövetés.

A MUPID—BASIC különlegességei a BASIC egyéb nyelvjárásaihoz képest (lásd *2. ábra*):

— speciális utasítások a vtx-központtal való információcseréhez,

— a CEPT grafikus lehetőségek támogatása.

Az egyéb BASIC segédprogramok a *3. ábrán* láthatók.

A MUPID-ba **M(UPID)CP/M** fejlesztőrendszer és operációs rendszer is betölthető. Ehhez az M—DISK mágneslemezegység szükséges, mivel ilyenkor a MUPID egy soronként 80 karaktert megjelenítő terminálként, az M—DISK pedig központi egységként működik. A felhasználó számára mindez egy 64 kbájtos CP/M operációs rendszer alatt futó számítógépként viselkedik. A programot a MUPID cég lemezen árulja, és az operációs rendszeren kívül megtalálható rajta a MUPID-ot 80 karakteres üzemmódba állító program, különböző adat- és fájlmásoló, -módosító és -törlő programok, valamint a lemezeket formázó és másoló program. Az MCP/M alatt futó és már széles körben rendelkezésre álló programok: a WORDSTAR, a MULTIPLAN, a DBASE II, a C—COMPILER, a PASCAL—COMPILER és a Z80 ASSEMBLER és LINKER. (Folytatjuk.)

JURENKA OSZKÁR

Az üzletekben egyre gyakrabban (és — igen lassan bár, de vitathatatlanul — csökkenő áron) jelennek meg a CD (compact disc) lemezjátszók vagy a már közkeletű nevükön: lézerlemezjátszók. Ezekről annyi közismert, hogy a hagyományos eljárással készült hanglemezeket nem tudják lejátszani, viszont az egészen más hangrögzítési technológiával előállított, kisebb méretű, „fizikai barázdákat" nem tartalmazó, újfajta lemezeket igen, és a hangminőségük, illetve a hangreprodukáló képességük sokszorta jobb a közönséges lemezjátszókénál

A jövő ígérete

Optikai tárolás ma és holnap

A látványos minőségjavulás kétségtelenül az új technológiának, a digitális hangrögzítésnek és a lézeres letapogatásnak (tehát a mechanikus komponensek nagymértékű kiküszöbölésének) köszönhető.

Jóval kevésbé köztudott azonban mind a mai napig az a tény, hogy CD-ket a számítástechnikában (jelenleg még szinte kizárólag a nagy információs rendszereknél) is eredményesen használnak nagy kapacitású, gyors elérésű háttértárolóként. A történeti hűség kedvéért meg kell jegyeznünk, hogy CD-k jóval előbb megjelentek a számítás-(illetve hadi-)technikában, mint a kommerciális elektronikai termékek sorában; ez esetben is ugyanaz játszódott le, mint már korábban annyiszor máskor (például néhány évtizede a kazettás magnóknál; a „compact" elnevezés eredete ugyanaz, mint a hagyományos audiokazetták C jelzése).

A számítástechnikában az „optikai forradalmat" jóval korábbra ígérték. Már a 70-es évek elején, amikor éppen hogy csak sikerült a lézert ipari keretek közé szorítani és elnyomni a szinte kizárólag „életveszélyes fegyver" hírét, megjelentek a kísérleti laboratóriumokban az optikai memóriák (holografikus tárolók). Ezek a koherens lézersugarak azon képalkotó tulajdonságát próbálták meg kihasználni, hogy egy kép bármely kis részlete hordozza a teljes képi információt. Ha ezt sikerülne megbízhatóan átúltetni a gyakorlatba, akkor igen nagy tömegű információ ma még szinte hihetetlenül hatékony tárolása és igen gyors visszakeresése válna lehetségessé. Ennek elméleti alapjai — hála Gábor Dénes Nobel-díjas tudósunknak — már léteznek. Az optikai forradalom azonban — a kétségtelenül létező eredmények, például az üvegszálas adatátviteli vonalak ellenére — késik. Lehet, hogy a CD az első igazi jele?

Nézzük meg a CD-ket számítástechnikai szemszögből. Mi is ez a CD? Egyszerűen egy 12 cm átmérőjű, 1,2 mm vastag műanyag lemez (közepén egy 1,5 cm átmérőjű pozicionáló lyukkal), amelyet fény irányába tartva a szivárvány minden színében pompázik. Ez a kis lapocska azonban 550 Mbájtnyi adatot képes hordozni, ami körülbelül 1500 hajlékonylemezzel (vagy körülbelül 250 000 átlagos könyvoldallal, egy nagyobbfajta enciklopédiával) egyenértékű. A mai CD egységekkel a lemezen tárolt bármely információ 2-3 mp alatt képernyőn megjelenithető.

A CD technológia

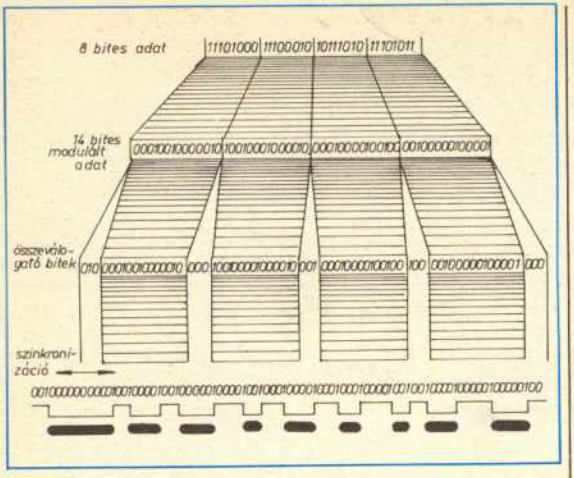
Az igen jó paraméterei ellenére a lemezek előállítása és az információ rögzítése meglepően egyszerű. Elöljáróban ki kell emelni, hogy a jelenleg elterjedt típusok ún. CD—ROM (compact disc—read only memory) tárolók, vagyis — ellentétben a hagyományos lemezekkel — az információt (például egy nagy adatbankot) megváltoztathatatlan formában, gyárilag rögzítik, azt törölni vagy felülirni nem lehet. Léteznek azonban már CD/I (compact disc interactive) egységek is, amelyekről a későbbiekben teszünk említést.

A CD-ken az információ speciálisan kódolt formában helyezkedik el: szemcsék (angolul *pit*-nek nevezik, ami hozzávetőlegesen felmaródást jelent) és sima részek (*land*) sorozataként. Először egy mesterlemezt hoznak létre (üvegből), melyen a körülbelül 0,6 μm széles és 0,12 μm magas szemcséket lézersugár készíti. A szemcsék közötti sima felület 0,9—3,3 μm hosszú lehet. Az információ — a hagyományos lemezektől eltérően — folytonos spirál alakjában helyezkedik el. E spirál „menetemelkedése", tehát a szomszédos „sávok" távolsága 1,6 μm, tehát a CD sávsűrűsége 16 000 sáv/inch, ami majdnem 170-szerese a hagyományos hajlékonylemeznek.

A mesterlemezről polikarbonát-műanyag lemezeket lehet szinte tetszőleges példányszámban nyomni: ez a felhasználói információhordozó. Ezt védő lakkréteg borítja, mögé vékony, filmvisszaverő alumíniumfilmet visznek fel.

A CD—ROM olvasása

A lemezről az információ GaAs (gallium-arzenid) lézersugárral olvasható vissza. A megfelelő tulajdonságú (fókuszált, pozicionált) sugarat optikai fej állítja elő. A visszavert fénysugár mennyisége jelzi, hogy a lemezen szemcsét vagy sima felületet érzékelt: a szemcse szétszórja a sugarakat, a sima felületről



nagyrészt visszaverődik az optikai fej irányába. A visszavert fényt egy fényérzékelő az intenzitással arányos erősségű elektromos árammá alakítja. Az árameltérés változtatható át aztán számítógéppel értelmezhető információvá.

Hibavédelem és -javítás

Ha az információhordozókat cseréljük, azaz az információt egyik megjelenési formájáról átalakítjuk egy másikra, megvan a veszélye annak, hogy tartalma torzul, hibásan vagy egyáltalán nem lesz értelmezhető. Ez a veszély természetesen fennáll akkor is, amikor a mágneses hordozón tárolt információt visszük át CD-re. Selejtes CD-k leggyakoribb okai a rosszul kialakított szemcsék, a fényvisszaverő anyag hibái, valamint a polikarbonát károsodása (megjegyzendő, hogy az egyszer már jól legyártott CD csak igen komoly fizikai hatás következtében sérül meg). Hibajavító eljárás nélkül a lemez fizikai elégtelenségei súlyos bithibákhoz vezethetnek, ezért a CD mesterlemez gyártásakor hibajavító kódolást alkalmaznak. A CD technológiában a kisebb anyaghiba igen valószínű, így a most ismertetendő javítóeljárás a selejtszázalékot nagymértékben csökkenti, tehát annak ellenére, hogy a lemezkapacitás ezzel majdnem 70%-kal csökken, a teljes CD technológia gazdaságossága növelhető.

Az információt korábban a papír- és mágneses hordozón, újabban a CD-ken olyan bitsorozat képviseli, amelyet ún. csatornakóddá kell konvertálni, ugyanúgy, mintha a biteket kommunikációs csatornán való átvitelre készítenék elő. A nagy kapacitású háttértárolók — beleértve a CD-ket is — bináris (azaz kétállapotú) csatornakódot használnak. A CD—ROM-ok esetében a csatornakódok a szemcsékkel és sima részekkel leírt információs „terület" kezdetét és végét jelzik.

A CD—ROM-gyártásnál használatos bináris csatornakódot EFM-nek (eight-to-fourteen — nyolcról tizennégyre — modulációnak) nevezik. Az EFM a felhasználói adatokat, a hibajavító adatokat, a címinformációkat és más szükséges kódokat csatornabit-sorozattá alakítja. A csatornabiteket bináris jelekké, majd pedig szemcsékké alakítják.

A CD egységekben lévő optikai olvasófej a modulációt ellenkező irányba hajtja végre, így visszaállítja az eredeti felhasználói és hibajavító információt.

Az *ábrán* látható, hogyan feleltethető meg bináris csatornakódoknak szemcsék sorozata a CD-n. A megszokott mágneses hordozókkal ellentétben a CD—ROM csatornakódokban az 1-esek és 0-k nem magát a digitális információt jelentik, hanem az 1-esek a változásra, a 0-k a változás hiányára utalnak.

Mivel két átmenet (két 1-es csatornabit) nem követheti egymást, a gyártók három szabványos csatornabitet hagynak a szomszédos átmenetek közötti legkevesebb elválasztáshoz. A szemcsék és sima területek sorozatának tehát a legnagyobb hossza 11 bit, és az egy bájt ábrázolásához minimálisan szükséges bitek száma 14. Ez a 14 csatornabit alkotja a CD—ROM szimbólumot. Végül három „összeválogató" csatornabitet használnak az egy adatsorozaton belüli szimbólumok összekapcsolására, tehát összesen 16 csatornabit szükséges minden felhasználói bájthoz. A lemezes információtárolásra jellemző szinkronizáció 24 (17 bites) szimbólumból áll, amelyet további 24 csatornabit és 3 „összeválogató" bit egészít ki. A CD—ROM alap információs egysége, az ún. keret úgy jön létre, hogy még további vezérlő és megjelenítésvezérlő, valamint hibajavító karaktereket illesztenek a bitsorozathoz. A CD—ROM keretinformációiról eddig elmondottakat a következőképpen foglalhatjuk össze:

adat	24 × (14 + 3)	csatornabit
szinkronizáció	24 + 3	csatornabit
vezérlés	1 × (14 + 3)	csatornabit
hibajavítás	8 × (14 + 3)	csatornabit
1 CD—ROM információs keret:	588	csatornabit

A lemez szervezésére ezenkívül még jellemző a blokk, amely 98 keretet tartalmaz. A fenti összefoglalásból megfigyelhető, hogy 24 felhasználói bájt tárolásához 588 csatornabit szükséges, tehát a lemez kihasználtsági foka valamivel jobb 30%-nál.

A CD/I

A CD technológiánál már említettük, hogy létezik ún. interaktív CD is: ezt nevezik CD/I-nek. Ez azonban nem olyan értelemben jelent interaktivitást, hogy a lemezre felhasználói módon írni is lehet. A CD/I elsősorban önálló használatra készül, nem számítógép perifériájának — sőt nem is elsősorban számítástechnikai célra.

A CD—ROM — mint említettük — előre rögzített, digitális információt tartalmaz. A CD/I hangot, képet, szöveget és programokat egyaránt tartalmaz — szintén meghatározott formátumban —, alapvetően házi használatra. A CD/I lejátszókhoz, amelyekbe meglehetősen bonyolult mikroprocesszoros vezérlő van beépítve, hagyományos erősítők és tévékészülékek is hozzákapcsolhatók, így egyszerre hallgathatunk zenét és játszhatunk tévéjátékot segítségükkel. Filmek tárolására azonban nem alkalmas, így a képlemezjátszót nem helyettesíti.

MARSCHIK IVÁN

Pályázati felhívások!

Lapunk 1987/10. számában közölt pályázati felhívásokhoz kapcsolódóan most részletezzük a jelentkezési feltételeket és tudnivalókat. Figyelem! A jelentkezési lapokat nem kell kivágni a Magazinból; az a lényeg, hogy a rovatokat megfelelően kitöltve küldjék el a megadott címre és határidőre.

JELENTKEZÉSI LAP
a μ'88 „Oktatóprogramok" versenyre

Név: .....

Levelezési cím: .....

Csapat esetén az iskola típusa: .....

A csapat neve: .....

A program(ok) adatai:

kategória: .....

neve(i) .....

a gép típusa: .....

programozási nyelv: .....

Bemutatási időigény: .....

Dátum: Aláírás:

Megjegyzés: a jelentkezési lappal együtt a program dokumentációját is kérjük.
A programot kazettán (többször felvéve) vagy hajlékonylemezen csak a bemutatóra kell elhozni. A bemutatón Primo, HT—1080Z, C16 és C64 gépek lesznek. Egyéb típusokra készített munkák elbírálásához kérjük, hogy a gépet hozzák magukkal. A beosztásról és egyéb részletekről minden időben pályázót írásban értesítünk. A pályázaton mindenki csak saját készítésű programmal vehet részt.

Jelentkezési határidő:
1988. február 15.
Cím:
NJSZT
„Oktatóprogramok"
1368 Budapest 5. Pf.: 240

JELENTKEZÉSI LAP
a μ'88 „Házi építésű számítástechnikai eszközök" versenyre

Név: .....

Levelezési cím: .....

Kategória: (rendszer/periféria/modell): ..

a rendszert (perifériát) tervezte: .....

a nyomtatott áramkört tervezte: .....

a dobozt tervezte: .....

a hardvert összeszerelte: .....

a hardvert bemérte: .....

a hardver rövid meghatározása: .....

Bemutatási időigény: .....

Egyéb: .....

Dátum: Aláírás:

Megjegyzés: a bemutatón program segítségével kell demonstrálni a hardver működését. Ehhez a hálózati csatlakozón kívül hozzák magukkal írásos hozzájárulásukat a kiállításon való bemutatáshoz.
A beosztásról és egyéb részletekről minden időben jelentkezőt írásban értesítünk.

Jelentkezési határidő:
1988. február 22.
Cím:
NJSZT
„Házi építésű számítástechnikai eszközök"
1368 Budapest 5. Pf.: 240

JELENTKEZÉSI LAP
a μ'88 „Sütés-főzés számítógéppel" versenyre

Név: .....

Levelezési cím: .....

A számítógép felhasználási módja: .....

A program milyen gépre és milyen nyelven készült: .....

Bemutatási időigény: .....

A bemutatáshoz a következőkre van szükségem: .....

A részletesebb leírást mellékletként küldöm: .....

.....

Dátum: Aláírás:

Megjegyzés: a beosztásról és a lebonyolítással kapcsolatos egyéb részletekről minden időben pályázót írásban értesítünk.

Jelentkezési határidő:
1988. február 22.

Cím:
NJSZT
„Sütés-főzés számítógéppel"
1368 Budapest 5. Pf.: 240

Tájékoztató a budapesti HCC (Homebrew Computer Club) klubokról

APPLE: Vezető: Ditrich Diebel
Hely: NJSZT — (V. ker., Báthori u. 16.)
Idő: minden hónap 1. és 3. kedd 15 órától

ATARI:
Vezető: Hirschler Tamás (534-988)
Hely: NJSZT — (V. ker., Báthori u. 16.)
Idő: minden hónap 1. kedd 17 órától

COMMODORE:
Vezető: Dr. Simonyi Endre (556-245)
Hely: TIT Stúdió — (XI. ker., Bocskay út 37.)
Idő: minden hónap 2. csütörtök 18 órától

HOMELAB:
Vezető: Barabási Rezső (277-880)
Hely: Belvárosi Művelődési Ház (VI. ker., Molnár u. 9.)
Idő: minden kedden 15 órától

PRIMO:
Vezető: Somogyi György (577-722)
Hely: Úttörő és Ifjúsági Ház Szabadidőközpont (VII. ker., Almássy tér 6.)
Idő: minden hónap 2. hétfő 18 órától

SINCLAIR:
Vezető: Hivessy Ferenc
1. Hely: Szellőzőművek (XI. ker., Építész u. 8/12)
Idő: minden hétfő 18 órától
2. Hely: GANZ Villamossági Művek MHSZ Rádió Klub (II. ker., Fény u. 11.)
Idő: minden hónap 2. szerda 18 órától

XT/AT:
Vezető: Dr. Bárkai János (452-755)
Hely: BME (F ép. II. szárny II. em. 9.)
Idő: minden szerda 15 órától

68XX:
Vezető: Vágh István (337-917)
Hely: TIT Stúdió (XI. ker., Bocskay út 37.)
Idő: minden hónap 1. kedd 17 órától

μ mikro számítógép magazin

Olvasóink kérésének eleget téve közöljük lapunk elmúlt évi számainak tartalomjegyzékét. Az alábbiakban címek előtt a lap- illetve oldalszámozás található. Végül rövid tartalmi leírás zárja a sort.

Tartalomjegyzékünk PROPER 16 számítógéppel készült. Az ékezetek hiányáért elnézésüket kérjük.

A szerkesztőség

1987. évi tartalomjegyzéke

ROVÁNZÓN KÍVÜL

1/7 Számitastechnika a fiataloknak*oktatas/informatika*Mongolia*Apple II-kompatibilis*bolgar modell alkalmazasa
1/9 Mit tud a PROLOG?*mesterseges intelligencia*alapismeretek*hazai fejlesztesi eredmenyek
1/15 A VIC-20 tarcisei*RND rutinok listaja
1/24 Nano a Snectrumon*abrite kezeles*programlista*alapelv es peldak
1/25 Uj karakterek*PTA 4000*programlista*felhasznaloi karakterek
1/27 VU-SZT megjelologram*ZX-Spectrum*grafika*programismertetes*alkalmazasi peldak
1/28 A viszolgaltatott ember*elektronizalas*tarsadalmi osszefuggesek*alkalmazasi peldak
1/36 Helyi halozatok*IBM PC/XT/AT*4-8 terminalos rendszerek*piaci kinalat
1/38 Fantazia es tudomany*Mandelbrot halmaz*PTA 4000*programlista
1/40 Baratunk a szamitogep*hattertar*adatbazis kezeles*tavfeldolgozas*felhasznaloi igenyek
1/43 Forum
1/45 PC Piaci prognozis
1/48 NCC 86
2/14 A programozas Trabantja*Basic*alkalmazasi elonyok*hatranyok
2/15 Vigyazat! Tolvaj!*Commodore 64*levedes*lrassurusag valoztatas*savformatum modositas*formazas
2/16 Hajozologepek*grafika*hazai gyartmanyvalasztek
2/18 SICOB - volt es lesz!*kiallitas*elozetes informacios
2/19 A DIGITAL rejtvenypalyazat eredmenyhirdetese
2/20 Acom a magyarazatot!*Primo*programozasi tippek*input es read
2/21 A Spectrum operacios rendszere ZX81-en*utmutato a Spectrum Basic-hez*ROM athelyezes
2/30 A Beta BASIC es alkalmazasa III.*ZX-Spectrum*programlista*magyar betukeszlet
2/32 Mandelbrot-program*Commodore 64*grafika*programlista
2/34 VC20 programok beolvasasa*Commodore 64*orautem illesztes*programlista
2/35 Floppy monitor*Commodore 64*lemezegyseg 1541*programlista
2/39 Interaktiv kompakt lemezek*Microware OS9*szabvanyositas
3/28 Vigyazat! Tolvaj!*Commodore 64*lemezegyseg 1541*levedes*blokkszerkezetek*hibafelismero GCR kod
3/30 Olvastunk... -Veglatek*sakk*programlista*Pascal-nyelvu program*alkalmazasi tapasztalatok
3/34 Rainbow - STYLIST-Epson nyomtato kezeloje*TRS-CoCo*nyomtatas*programlista*reprint a Rainbow magazinbol
3/41 Ki ad magyarazatot?
3/43 Egy szamitastechnikai-gazdasagi konszern: a COMPUTERMALL
3/43 Forum
4/10 Tavkonferencia*fejlodestortenet*helyzetkep
4/12 Vegyem, ne vegyem?*SUGAR-16*rangsorolo porogram*kereskedelem*termekminositas*gepkocsi kivalasztasi pelda
4/13 Az ev szamitogepe
4/14 Emlekkepek Neumann Janosrol
4/21 Mit tud a Pascal?*definialt adattipus*utasitasok*eljarasok*fuggvenyek
4/24 Rainbow - MS-DOS konvertalas CoCo lemezre*TRS CoCo*fajltranszformacio
4/31 Mikrogepes uttorok*iskolaszamitogep*Kina*utiteszamolo
4/32 Vigyazat! Tolvaj!*levedes*lemezegyseg 1541*PDC-hibalista*21-es hiba felhasznalasa
4/37 Az Atari 800 XL*altalanos ismertetes
5/10 PC utanzatok a piacon*IBM PC*klonok*kompatibilis gepkinalat
5/12 Tavkonferencia II.*mukodo rendszerk bemutatasa*trendadatok
5/16 Rainbow - CoCo konvertalas MS-DOS lemezre*TRS-CoCo*programlista
5/18 Mesterseges ertelem*mesterseges Intelligencia*keresesi strategiak*problema-nyilvantartas
5/27 Vigyazat!Tolvaj!*Commodore 64*lemezegyseg 1541*levedes*2O-as tipusu hiba generalasa
5/28 A szoftvergyartas nagyjai*Microsoft*MS-DOS*interju Dr. Simonyi Karollyal
5/33 Printerszalag hazilag*csere hazilag*olcso irogepszalag hasznalata
5/35 Z80 szimulator*gepi kodu programokhoz
5/36 Adatbazis II.*szovegfeldolgozas*integralt szoftver*Dragon-Basic*programlista
5/38 Olvastunk... - Visszalepeses algoritmusok*sakk*Commodore 64*programlista
6/3 Nemes Tihamer Orszagos Kozepiskolai Tanulmanyi Verseny
6/9 Hazi- es iskolaszamitogep-merleg*piaci kinalat*hazai arak
6/11 ATARI kontra COMMODORE 64*osszehasonlitas
6/15 Mesterseges ertelem II. - Alkalmazott intelligencia*mesterseges intelligencia*Prolog*szakertoi rendszerek*VAX 11/780
6/24 Kritika es onkritika*helyesbites*87.01.15.oldal
6/25 Mikro 87*kiallitas*beszamolo
6/34 Vigyazat! Tolvaj!*Commodore 64*lemezegyseg 1541*levedes*22-es tip. hiba generalasa*DBID modositas
6/36 Olvastunk... - Az adatok es a valosag*informatika-elmelet*adatstrukturalas*jelensegleiras
6/42 Einsteinnel vitazva
7/10 ATARI kontra COMMODORE 64*osszehasonlitas
7/21 Ipari park a muegyetem mellett*Gorogorszag*haziszamitogep*ertekelesi halozat, arak
7/22 Vigyazat! Tolvaj!*levedes*Commodore 64*lemezegyseg 1541*29-es tipusu hiba eloallitasa
7/23 Mesterseges ertelem III. - Reprezentaciok*mesterseges intelligencia*szakertoi rendszere*allapotter definialas
7/27 Az adattipus fejlodese I.*szoftvertudomany*fejlodestorteneti attekintes
7/29 Tavcitrula a Balatonban*kiallitas*magyar muszertechnika*gyogyszeripar*ZX-Spectrum
7/38 Olvastunk... - Operacios rendszerek*konyvismertetes
7/41 Meditacio*felhasznaloi rendszer*hazai szakmai hianyossagok
7/44 Aprocska*oktatas/matematika/jatek*programlista*buveszkedes
8/8 Futyul a magyar szamitogepes tanulas vonata*oktatas/informatika*oktatoprogram*iskolaszamitogep*hazai helyzetkep
8/15 ATARI kontra COMMODORE 64*lemezegyseg 1050, 1541*osszehasonlitas*tomb es fajlkezeles
8/26 Vigyazat! Tolvaj!*levedes*Commodore 64*lemezegyseg 1541*programlista
8/27 Mesterseges ertelem IV. - Tanulas*mesterseges intelligencia*szakertoi rendszerek*analogias tanulas*informacio felhalmozas
8/31 Hallaskolvasas*Brailab*CPK1*vakok rehabilitacioja*alkalmazasi peldak*fejlesztesi tervek
8/32 Az adattipus fejlodese II.*szoftverelmelet*fejlodestortenet 2*programnyelv
8/42 Elektronikus lemez*hattertar*RAM-floppy*hazai kinalat, arak
8/44 KANYAN A HARSO!*grafikus adatbazis*terkepes nyilvantartas*termekismertetes
8/45 Petike szetszedi apuka orajat*oktatas/matematika*jatek*oraleolvasasi trukkok
8/48 Vasari kitekinto*tavaszi BUV*Kodex 2000*szovegszerkesztes*kiallitas*beszamolo
9/13 Ne feljunk, meghalalja*oktatas/szamitastechnika*iskolai alkalmazas*Szolnok*tapasztalatok,tervek
9/14 ATARI kontra COMMODORE 64*adatbevitel,lanyvezetes*programlista*osszehasonlitas
9/22 Vigyazat! Tolvaj!*levedes*Feltores*formazas*KIMT
9/24 Mesterseges ertelem V. - Gepi ertekeles*mesterseges intelligencia*kepfelismeres*alakfelismeres
9/27 Mit tud a SYMPHONY?*informaciokezeles*szovegfeldolgozas*IBM PC/XT/AT*ismertetes*alkalmazasi peldak
9/35 Abrakeszites eloadasokhoz, cikkekhez*dia*fotozas*Sinclair Spectrum*programlista
9/38 Olvastunk... - Mi a mesterseges intelligencia?*digitalis technika*konyvismertetes
9/42 Petike oroknaptart keszit*oktatas/matematika*programlista
10/22 Mit tud a SYMPHONY? II.*integralt szoftver*informacio feldolgozas*IBM PC/XT/AT*ismertetes 2
10/26 Segit a vonalkod*kereskedelem*felhasznalo rendszere*Skala Metro*alkalmazasi tapasztalatok
10/27 Cora es a szamitogep*Genie II.*programcsomag adaptalasa
10/28 Milyen az intelligens videotex-dekoder? I.*MUPID*posta*hazai tervek
10/31 Az IBM segit robbant*IBM PS/2*oikroszamitogep csalad*CHIP teszt ismertetese
10/32 Szemuveg alatt a szamitastechnika artalmai*ember-gep kapcsolat*vedoszemuveg*termekismertetes CompuDrug
10/34 Jar nec bunda a bunda...*sport*alkalmazasi pelda*jatekvezeto-kijeloles
10/40 Olvastunk... - Az informacio mennyisege*gondolatok az informaciorol
11/24 Milyen az intelligens videotex-dekoder? II.*MUPID*vtx-dekoder belso felepitese*vtx-dekoder mukodesi modja
11/26 Tyosztok szalnaplo...*tipusvalasztek bemutatasa*piaci trend
11/28 Kenyszerfejlesztesi eszkozok - Editor*szovegszerkesztes*ember-gep kapcsolat*megkivant jellemzok*mukodesmod, parancstipusok

11/29 Izgalmas kezdet*oktatas informatika*szamitogepes tabor*Almassy-teri kezdemenyezes
11/30 Meg egyszer az Enterprise-rol*ismertetes kiegeszitese
11/38 Az Olivetti nem hibazik - CPS*biztonsag*hibaturo rendszer ismertetese
11/40 Olvastunk... - Szamitogeppel megoldhato feladatok*filozofia*Turing-fele gep*a szamitogep fogalma*definiciok
11/45 Petike kartyazik*jatek*programlista*zsirozas
11/44 A termeszet vigyaz rank*SCI-FI*novella
12/3 Atadtak az 5000. Videoton iskolaszamitogepet*Kopeczi Bela
12/23 Z88, az uj Sinclair csoda*ujdonsag bemutatasa
12/24 Ajandekajanlat*mikroszamitogep piac*hazai kinalat*arak
12/26 Rendszerfejlesztesi eszkozok II. - Assembler*forditoprogram*programozasi alapfogalmak*mukodesmod elemzese
12/28 Milyen az intelligens videotex-dekoder? III.*MUPID*dekodertipusok*dekoder uzemmodok
12/39 Orankent huszert*szamitogepes klubok helyzete
12/40 Olvastunk... - Egy Updike regeny nyoman*konyvismertetes*gondolatok a ket kulturarol
12/45 Petike adot szamit*oktatas/matematika*jatek*programlista

ISKOLA-SZAMITOGEP

1/3 Kernal rutinok 4.*Commodore 16
1/3 C16 orgona*zene*programlista
1/4 Spectrum az asztalban*ZX-Spectrum*barkacsolas
1/4 Terkep*Primo*oktatas/foldrajz*programlista
1/5 Kozepiskolai Szamitastechnikai Konferencia*Fonyod*oktatas/informatika*beszamolo
2/3 Meghalt a kiraly, ki lesz az uj kiraly!?*iskola*TII*Videoton*Commodore*ajanlott geptipusok
2/4 Ismeros gondok az Egyesult Allamokban*iskola*alkalmi helyzetkep*tapasztalatok es tervek
2/8 Primo - primatus*kiegeszites a felhasznaloknak*roviditett Basic nevek
2/8 Hasoloprogram*Primo*programlista
2/8 Gepi kodban*Primo*programlista*ertekadas*adatatadas
3/5 Folyekony logika*iskola*oktatoprogram
3/5 Gepi kodban*Primo*programlista*cimzes keszites
3/6 Megszakitasvezerelt digitalis ora*ZX-Spectrum*programlista
3/9 Pontszeru toltesek erovonalkepe*PTA 4000*oktatas/fizika*programlista
4/3 Felhasznaloi tokenek*Commodore 16*interpreter tokenizalo rutin
4/5 Egy uj Basic forditoprogram: a PLUS-Comp*Commodore 16,116,64,Plus 4*hardver igeny
4/6 Gepi kodban*Primo*jatekprogram keszitese*vezerlo billentyuk atalakitasa*programlista
5/3 Elmeny a noveny: szimulacios modell a kozepiskolaban*oktatas/biologia*Commodore 64
5/5 Szovegszerkeszto*HT-1080Z*programlista
5/7 Adatatadas Basic-bol gepi kodba*Primo*programlista*kepernyopont ki/bekapcsolas*programfutasi ido merese
7/2 Memory suli*Commodore Plus 4*oktatas/tortenelem*kerdes-felelet*kartyacsomag hasznalat
7/5 Gepi kodban*Primo*grafika*programlista*negyszograjzolas
8/3 Bemutatjuk az OPI programirodajat*oktatas/informatika
8/4 Egy egeszen egyszeru vilagmodell*nepessegvizsgalat*statisztikai modellezes*elmeleti alapok*alkalmazasi peldak
8/6 A Nemes Tihamer OKSZTV dontoje 1987*versenyertekeles
8/6 Nemes Tihamer*megemlekezes
9/4 Tanulni, tanulni, tanulni!*oktatoprogram*beszerzes*hazai kinalat/arak
9/6 Fonalfek, fonalero*Sinclair Spectrum*programlista*textilipar*automatizalas
10/3 Oktatok-valtozo szerepben*oktatas/informatika*beszamolo a tapasztalatokrol
10/6 UNICOMP vagy TechnoMIR*illesztorendszer*iskolai alkalmazas
10/7 A NESTOPSI titka*oktatas/orosz nyelv*Commodore 64*ZX-Spectrum
10/11 Nemzetkozi szamitastechnikai verseny Bulgariaban*beszamolo
11/3 MINI PLUS-Comp Basic fordito*Commodore Plus 4*kazettarol toltheto valtozat
11/2 A ROM-ban tarolt aritmetikai rutinok hasznalata*TV-Computer*gepi kod
11/5 TechnoMIR*illesztorendszer*iskolai alkalmazas*mukodesmod*modulvalasztek
12/4 A ROM-ban tarolt aritmetikai rutinok hasznalata*TV-Computer
12/6 TechnoMIR - DIN modul*illesztorendszer*a digitalis input modul DIN

DIAKROVAT

1/8 Karakterkeszlet editor*Commodore 64*programlista
2/10 Tv-foci*HT-1080Z*jatek*programlista
3/11 Hibauzenetek olvasasa*Commodore 64*lemezegyseg 1541*programlista*DOS-hibauzenet
3/12 Terhatasu abrak*ZX-Spectrum*programlista
4/6 Megszakitas*Primo*sprite kezeles*peldaprogramok
4/8 Eljarasdefinialas*Commodore 16*strukturalt programozas*programlista*nevvel hivhato szubrutin
4/9 Betuzsonglorkodes*Primo*programlista*diszito feliratozas*karakterkiiro szubrutin
5/9 Rajzoloprogram*ZX-Spectrum*grafika*programlista
5/10 Karakteres kepernyot masolo rutin*Commodore 64*nyomtatas*programlista
6/5 Kepernyosurito*ZX-Spectrum*programlista
6/6 Szalagujsag*ZX-Spectrum*programlista
6/6 HT Magyarul-hibaztal!*HT-1080Z*hibauzenet tablazat
6/7 Autostart kazettara*Commodore 64*levedes*kazetta*RUN-STOP/RESTORE hatastalan
7/6 Nyujtott karakterek*Commodore 64*programlista
7/6 Finom kiiratas Primora*programozasi tippek
7/7 Grafika*Commodore 16,64*Simons Basic*programlista*matematikai fuggvenyek
7/8 Harom rutin*ZX-Spectrum*programlista
8/10 Ekezetes karakterkeszlet nyomtatora*Commodore 64*programlista
8/11 Ketnapi tizorai*Commodore 64,Plus 4*szovegszerkesztes*fenyszedes
8/12 Aprosagok*ZX-Spectrum*grafika*Mandelbrot halmaz*programlista
8/12 Szintetizator*ZX-Spectrum*zene*programlista
9/9 Rajzoljon Spectrummal!*Sinclair Spectrum*programlista*grafika
9/10 Periferiateszt C128-ra*programlista*periferia-keszenlet ellenorzes
9/11 Utkereses grafban*algoritmus*programlista*Commodore 64
10/12 Magyar nyelvu Basic C64-re*programlista
10/13 Spectrum reflex*ZX-Spectrum*programlista*reakcioido merese
10/13 Spectrumra irt binaris atalakito*ZX-Spectrum*programlista*karakter-bitterkep kiiratasa
10/14 Mastermind C64-re*Commodore 64*jatek*programlista
10/15 Tobb szolam Homelabra*zene*programlista
11/6 Barazdak*oktatas fizika*programlista*szimulacio
11/6 Betolto*Simons Basic*programlista
11/7 Energiaatadas*programlista*oktatas*fizika
12/3 Sprite-ok bonckes alatt*Commodore 64*sprite kezeles*programlista*pozicionalas kereten kivulre
12/5 Visszatekintes*szamitogepes tabor*beszamolo
12/9 Kepernyoeditor*jatekprogram keszites*programlista
12/11 Trukk Spectrumra*ZX-Spectrum*levedes

PROGRAMOZASTECHNIKA

1/12 Z80 programozasi gyakorlatok 6.*maszkolhato megszakitasok
1/14 Basic es gepi kod*Commodore 16,64,VIC20*kozvetett JMP utasitas
1/15 Adalek a rendezeshez*Commodore 64*programlista*tombepites
1/16 Binaris keresoalgoritmus C64-re*programlista*rendezett adathalmazok
1/17 Papirtakarekosites*ZX-Spectrum*nyomtatas*programlista*valtozo karaktersuruseg
2/12 Z80 programozasi gyakorlatok 7.*gepi kod*binaris fa listazasa
2/13 Basic es gepi kod*Commodore 16,64, VIC20*felteteles ugras*relativ cimzes
3/13 Overlay-Chain-Append I.*Commodore 64*levedes*atfedestechnika*titkosito modul beepites
3/12 Basic es gepi kod*Commodore 16,64*indexelt cimzes
3/16 Tombok kezelese FORTH-ban*gepfuggetlen tombkezeles
4/18 Basic es gepi kod*Commodore 16,64, VC20*indirekt/indexelt cimzes
4/15 Overlay-Chain-Append II.*Commodore 64*programlista*gepi kodu program betoltese
5/23 Basic es gepi kod*Commodore 16,64,VC20*allapotregiszter-bitek
5/24 Teknocbeka-grafika Basic-ben*Logo*Basic*programlista
5/26 Merge*Commodore 64*programszerkesztes*programlista
6/19 Programozzunk Pascalban: scroll*ZX-Spectrum*programlista
6/22 Basic es gepi kod*Commodore 16,64,VC20*kazettaegyseg*szalagrairas masodlagos cimmel
6/23 Basic igenyesnek, extraxtal eladó...*alkalmazhatosagi korlatok*kritikai eszrevetelek*vitaanyag
6/29 Tenudbor*Commodore 64*programlista*programfajl atsorszamozas

6/31 Teknosbeka-grafika II.*nyomtatas*merettorzitas kikuszobolese
5/32 Folyamatabrak*programtervezes*jelolesi lehetosegek*peldak
7/14 Basic es gepi kod*Commodore 16,64,VC20*futasi idok elemzese*osszehasonlitas a Basic-kel
7/15 Z80 programok haladoknak - Gyors LDIR*ZX-Spectrum*Primo*Z80*programozasi tippek
7/16 Teknosbeka-grafika III.*torlesi parancsok*pozicionalasi parancsok
7/17 Adatbeolvasas Pascalban*ZX-Spectrum*HP4T*forditoprogram*adatbeolvasas konnyitese
8/15 Basic es gepi kod*Commodore 16,64,VC20*leptetoutasitasok
8/18 Pontosabb szamitasok Pascalban*ZX-Spectrum*programlista*lebegopontos aritmetika*pontossaganak novelese
8/21 Z80 programok haladoknak - Manokezelo*Primo*ZX-Spectrum*sprite-kezeles*programlista
8/22 Teknosbeka-grafika IV.*rajzolo Basic rutinok
8/24 ...Babel tornyara*programnyelv*osszehasonlitas nehezsegei*objektiv ertekeles nehezsegei
9/24 Toronyepitat lanccal...*programnyelv*nyelvertekelesi szempontok
9/17 Basic es gepi kod*Commodore 64;VC20*programlista*logikai utasitasok
9/18 Z80 programok haladoknak - Illusztracio manokezelohoz*Primo*Sinclair Spectrum*programlista*grafika*sprite kezeles
9/20 Formazott listazas I. - Listazoprogram*Commodore 16;64;plus/4*formatalas*programlista*listanyomtatas
10/17 Z80 programok haladoknak - A Paint rutin*ZX-Spectrum*Primo*grafika*programlista*zart alakzat festése
10/17 Basic es gepi kod*Commodore 16,64,VC20*logikai utasitasok BIT es NOP
10/21 Formazott listazas II. - Osszegkepzes*Commodore 64*programlista*ellenorzo-osszeg generalasa 1.
11/12 Basic es gepi kod*Commodore 16,64,VC20*veremkezelo utasitasok
11/13 Z80 programok haladoknak - Festes mintaval*ZX-Spectrum*grafika*programlista*minta rajzolas, festes
11/14 Verem a Spectrum Basic-hez*ZX-Spectrum*programlista
11/15 A Spectrum masoloi, masolok spectruma*ZX-Spectrum*Zotyocopy*Copier FM3*Omnicopy 2*Copy-Copy*alkalmazasi tanacsok
11/17 Szamitogepes grafika Pascalban - egyenes*ZX-Spectrum*Hisoft Pascal*programlista*hianyzo grafikai lehetosegek
11/20 Hangos billentyuzet*Commodore 64*programlista
11/21 Lebegopontos tombok atadasa Basic - Pascal*programlista*atteres Basicrol*lebegopontos tombok formalasa
12/12 Basic es gepi kod*Commodore 16,64,VC20*eredeti cimre valo toltes*LOAD*GOTO rutin roviditese
12/13 Z80 programok haladoknak - I/O rutinok*ZX-Spectrum*Primo*programlista*sprite-rajzolas*kepernyokezeles
12/19 Szamitogepes grafika Pascalban - kor*ZX-Spectrum*programlista
12/21 Fuggvenyek es utasitasok*programozaselmelet*alapfogalmak*eljarasok,

µPROGRAMOK

1/22 A Beta BASIC es alkalmazasa II.*ZX-Spectrum*oktatas/angol nyelv*szotarkeszites
2/36 Disassembler program*PTA-4000*programlista*ROM-rutinok visszafejtese
2/19 Hazi modositasok a Gens3,Mons3 programokon*programlista*ekezetes uzemmod
3/24 Kalkulacios program adoelszamolashoz*ZX-Spectrum*VU-CALC*programlista*jovedelemnyilvantartas*adoelszamolas
3/25 Landobas*Commodore 64*nyomtatas*programlista*tablazatkiiratas
3/25 Hasznos rutinok C64-re*programlista*kurzorpozicionalo rutinok*szinvalto rutinok
3/26 Programbetoltes soros illeszton keresztul*Commodore 64*programlista
3/26 A ketutemu motor mukodese*ZX-Spectrum*oktatas/fizika*programlista
3/27 Haromdimenzios abrazolo geometria*ZX-Spectrum*oktatas*grafika*atrajzolas egysikura
4/26 Menuk*ZX-Spectrum*programlista*altalanos menu-rutin
4/26 Oszlopdiagram-rajzolo*ZX-Spectrum*grafika*programlista
5/30 Disassembler program*Commodore VC20*programlista
5/32 Kulonleges input*Commodore 64*programlista*menuvalaszto billentyuk
7/30 Hasznos programok az USR fuggveny alapjan*Commodore 64, VC20*programlista*modositas szalagfajlcserehez
7/31 Piszok ellen sortorlo*ZX81*programlista
7/32 Szinvaltas*Commodore 64*programlista*szinvaltas funkciobillentyuvel
7/32 Relativ adattarolas*Commodore 16*programlista*relativ fajlok kezelese*automatikus szamformatumozas
8/34 Magyar ekezetes karakterkeszlet EPSON nyomtatokra*nyomtatas*programlista*karakterkeszlet Epson FX-hez
10/33 Beta Basic*ZX-Spectrum*programlista*fuzerek pozicionalt beirasa
10/33 Korszerubb - APPEND C16-ra*programlista*programkapcsolo
11/31 Felhasznaloi karakterek*ZX-Spectrum*grafika*programlista
11/31 Memoriaolvaso program*Commodore 16,64,VC20,Plus 4*programlista*tarcim-kiiratas hex formaban
12/31 Datakonverter*Commodore Plus/4*programlista*ellenorzoosszeg eloallitas*data sor begepeles
12/32 A MASTERFILE magyar valtozata*ZX-Spectrum*programlista*A VER 06 valtozat magyaritasa

µKLUB

2/37 HELP bovito*Help+*Commodore 64*programlista*ROM alatti RAM-ok vizsgalata
3/36 ZX-Spectrum-helyi halozatban*illesztes*atviteli eszkozok
3/37 ZX-Spectrum mint digitalis jeleket tarolo oszcilloszkop*programlista
3/39 Kulso ROM C64-hez*barkacsolas*Basic Interpreter alatti 8K
3/39 A VC20 hardver adta lehetosegei*grafika*zene*programozasi fogasok
3/40 The Users Port*menuprogram
4/32 Az adatbaziskezelo rendszer*szovegszerkesztes*Dragon Basic*integralt szoftver*programlista
4/35 The Users Port*Commodore 64*Easy script*Basic programhibak azonositasa*szovegszerkeszto szoftverekkel
4/36 Adom a magyarazatot!*nyomtatas*grafika*szamitasi pontatlansagok
5/34 The Users Port*lemez/hajlekony*tipusvalasztek*kezelesi tanacsok
6/38 Szovegszerkeszto I.*Dragon Basic*integralt szoftver
6/41 The Users Port*Commodore 64, 128*dec/hex-tablazatok
7/34 Szovegszerkeszto II.*Dragon Basic*integralt szoftver*atiras mas gepre
7/35 A VIC-1541 lemezegyseg furcsa titkai*levedes*Commodore 64*programozasi trukkok
8/26 Postazo I.*szovegszerkesztes*integralt szoftver*Dragon Basic
8/38 Egyszeru, Z80 alapu mikroszamitogep*barkacsolas*mikroszamitogep keszites*kapcsolasi rajz*szerelesi utmutato
8/40 Grafikus nyomtatas K6311 matrixnyomtatovel*Primo*nyomtatas*grafika*programlista*monitorkepmasolas K6311-re
8/42 Ket erdekes rovid program C64-re*programozasi trukkok
9/32 Az Enterprise 128*teszteles*osszehasonlitas*Atari,Commodore,Sinclair
10/36 Integralt szoftver*szovegfeldolgozas*Dragon Basic*programlista
10/38 Ket botkormannyal*Primo*barkacsolas*kapcsoloaramkor
11/32 2-4 kbajtos EPROM egeto a Spectrumhoz*ZX-Spectrum*barkacsolas*keszitesi utmutato*programlista
11/36 Fajlok osszerendelese*szovegszerkesztes*postazas*integralt szoftver*Dragon Basic
12/35 Postazo II.*szovegfeldolgozas*integralt szoftver*postazas*tablazatkeszites
12/38 Rejtett fajlok*IBM PC*programlista

SAKKPROGRAMOZAS

1/33 Allasertekeles VIII.*sakkprogramozas*tamado es vedekezo strategia*tablasulyozas
1/34 Chess Challenger*sakkszamitogep*Fidelity*Chess Challenger*tipusvalasztek ismertetese
2/38 Megnyitaselmelet*sakkprogramozas*megnyitasi konyvtar*algoritmus*poziciotabla
3/42 Dinamikus ertekelofuggveny*sakkprogramozas
4/43 Gyalogszerkezet*sakkprogramozas
5/42 Vegjatekok I.*sakkprogramozas
6/44 Bastyavegjatekok*sakkprogramozas
7/42 Futovegjatek*sakkprogramozas
7/43 A mikroszamitogepek 6. vilagbajnoksaga*sakk*beszamolo
8/43 Utescsere - algoritmus*sakkprogramozas
9/41 A jatekfa es kiertekelese*sakk*algoritmus*negamax eljaras*programozasi modszer
10/42 A jatekfa es kiertekelese 2.*sakkprogramozas
10/43 Az Elite csalad*sakkszamitogep*Fidelity*uj modellek ismertetese
11/42 A jatekfa es kiertekelese 3.*sakkprogramozas*fel-lepesek sorbarendezese*alfa-beta algoritmus
12/44 A jatekfa es kiertekelese 4.*sakkprogramozas

JATEKPROGRAMOK

2/40 Ket oras jatekok - miert ne?*Commodore 64*jatekprogram keszites*idomero
2/41 Labirintus*Laser-210*jatek*programlista
2/42 Kapcsolta??*matematika*programlista*Lagrange-interpolacio
3/45 Jatek Poke-ok*Commodore 64*orokelet-poke lista
4/46 Vilagosság a Setet Torony korul*Commodore Plus/4*leiras
4/46 Urhajozas*ZX-Spectrum*programlista
5/42 Csipp-csepp*ZX-Spectrum*programlista
5/44 Aprosagok*Commodore 64*jatekprogram keszites*nehany otlet kezdoknek
5/44 Szellem ellen szellemesen*Commodore 64*jatek*programlista*sprite-torles

Szerviz a város peremén...

Ellátogattunk a Professional Országos Számítógépszervizbe, ahol Gottfried Tibor, a lakossági szolgáltatásokat végző részleg vezetője az Enterprise-tulajdonosok egyre népesedő táborának szóló hasznos tanácsokat adott. Ezeket mi is olvasóink figyelmébe ajánljuk, mielőtt hibás(nak vélt) gépükkel a szervizbe rohannának.

Billentyűzet és botkormány

Talán sokan nem is tudják, hogy ha óvatosabban bánnak a kényesebb mechanikai részekkel, és az elektronikára vonatkozó néhány szabályt betartanak, rengeteg boszszúságtól kímélhetik meg magukat.

A gép használata során a billentyűzet és a botkormány van a legnagyobb igénybevételnek kitéve. Az Enterprise 128 billentyűzetének, az ún. fóliabillentyűzetnek egyik nagy előnye, hogy kis helyen elfér; ennek köszönhető a gép lapos kivitele. Hátrányai közül kiemelendő, hogy óvatos kezelést igényel. Ez persze nem azt jelenti, hogy úgy kell vigyázni rá, mint a hímes tojásra, és minden billentyű lenyomása előtt meg kell gondolni, nem fölöslegesen ütjük-e le. Csupán arról van szó, hogy ha kíméletesen bánunk a billentyűkkel, nem nyomjuk őket túl erősen, sokkal kisebb a valószínűsége annak, hogy tönkremennek. A gép egyébként minden billentyű leütését hangjelzéssel nyugtázza; így tájékoztatja a felhasználót arról, hogy nem kell erősebben nyomnia a billentyűt, mert már érzékelte azt.

A botkormánynak még nagyobb igénybevételt kell elviselnie, hiszen nemcsak a kurzor, hanem a legtöbb játék is azzal irányítható. Gyakori eset, hogy az ember a játék hevében a kelleténél erősebben tolja a botkormányt; ez szinte természetes is. Az is érthető, hogy sokan szeretnék a gépet játékra is használni. Aki teheti, külső botkormányt használjon a játékhoz, mert azt kifejezetten erre a célra készítették, és sokkal jobban bírja a strapát.

Amit a magnóról tudni kell

A géphez tartozó magnó sem a legmasszívabbak közül való. Már a kazettacserét is jó, ha óvatosan végezzük, de a magnó legérzékenyebb pontja a számláló, ezen belül is ennek nullázógombja; elég könnyen törik.

Finoman kell bánni a magnó és a gép csatlakozóival is. Volt már rá példa, hogy valaki olyan erősen nyomta a gépbe a magnó csatlakozóját, hogy az letört a panelről. Ezeknek a csatlakozóknak nincs más felfüggesztésük, csak az a három forrasztás, amivel a panelhoz kapcsolódnak. Ezért ha mód van rá, ne bolygassuk a már összeállított konfigurációt, és lehetőleg minél ritkábban nyúljunk a csatlakozókhoz.

Hogyan csatlakozzunk?

A csatlakozókkal kapcsolatban rá is térhetünk a legfontosabb elektronikai szabályokra. Nagyon fontos tudnivaló, hogy a csatlakozókat csak akkor szabad kihúzni vagy bedugni, ha a gép ki van kapcsolva. A vezetékekben ugyan normál állapotban nincs túl nagy feszültség, de előfordulhat, hogy az ember elektromosan feltöltődik, és ez a töltés a gépbe jut. Ilyenkor ott a megengedettnél nagyobb feszültség keletkezik, amely könnyen eljuthat a bonyolult integrált áramkörökig, és tönkreteheti azokat.

A tápegységről

Ügyelni kell arra, hogy a tápegység szabadon legyen. Ne rakjunk rá semmit (például könyvet), ami gátolná a hűtését! A tápegységet úgy tervezték, hogy a lehető legkevésbé melegedjen, de a hőfejlődés így is elkerülhetetlen.

Ahhoz, hogy jó minőségű és zavarmentes képet kapjunk, a gépet a tévétől legalább másfél méterre kell elhelyezni. Ez egyébként egészségügyi szempontból is kívánatos, hiszen aki túl közelről nézi a tévét, rontja a szemét.

Vigyem, ne vigyem?

Persze előfordul, hogy a leggondosabb bánásmód ellenére is meghibásodik a gép. Ha viszont elromlik, nem kell feltétlenül az egész berendezést elcipelni a szervizbe; sokszor elég, ha csak a hibás egységet visszük el. A hiba helyét úgy a legkönnyebb megtalálni, ha egy másik konfiguráció is rendelkezésünkre áll, és egyenként cserélgetjük a különböző egységeket és kábeleket. Ha erre nincs módunk, akkor a hiba jellegéből indulhatunk ki. Például ha bekapcsolás után a képernyő nem változik meg, és a gép LED-je sem jelzi a bekapcsolt állapotot, akkor minden valószínűség szerint a tápegység hibás. Ha bekapcsolás után megjelenik a megszokott kép, és a tesztelés után a villogó ENTERPRISE feliratot is látjuk, de egy billentyűt lenyomva nem jelentkezik be a BASIC, akkor biztosan a cartridge rossz. A hibás csatlakozót vagy vezetéket is megtalálhatjuk, ha ezeket óvatosan megmozgatva, változást észlelünk.

Aranyszabály, hogy ha valamilyen funkció nem működik, akkor a gép használója először magában keresse a hibát.

Legtöbbször a magnóról való betöltéssel van probléma. Ha a gép nem tölti be a programot a kazettáról, a hibának többféle forrása lehet. Gyakran csak a magnó feje piszkos, ezért tisztítsuk meg alkoholos ruhával. De más bajok is lehetnek a magnetofon fejével, mert nem mindegy, hogy hogyan áll; helyzetét a magnó előlapján található fejbeállító csavarral változtathatjuk. A módszer a következő. Betesszük a beolvasandó kazettát a magnóba és elkezdjük lejátszani. Közben kis csavarhúzóval csavargatjuk a fejbeállító csavart. Akkor jó a fejállás, amikor a legélesebb, legmagasabb hangot halljuk. Aki még soha nem csinált ilyet, csak hozzáértő személy jelenlétében nyúljon a csavarhoz!

Előfordulhat, hogy a tévé okozza a töltési hibát; vannak ugyanis olyan típusok, amelyekkel a gép nem hajlandó tölteni. Ezt egyszerű tévécserével megállapíthatjuk, vagy ha nincs még egy tévénk, akkor meg kell próbálni a betöltést a tévé kiiktatásával. Ha így sikerül, akkor biztosan azzal van a gond. Ilyenkor csak a gépet kell bevinni a szervizbe, ahol egy kis átalakítással megszüntetik a hibát.

A szervizben az átlagos javítási idő három-négy munkanap. A javítást végző szakemberek készséggel válaszolnak mind a hardverrel, mind a szoftverrel kapcsolatos kérdésekre.

ROMVÁRI GÁBOR

SZÁMÍTÁSTECHNIKAI KISSZÖVETKEZET

termelőszövetkezetek,
állami gazdaságok,
iparvállalatok,
városgazdálkodási vállalatok,
tanácsok
és
egyéb gazdálkodó szervek

VÁLLALJUK

ügyvitelszervezési,
rendszerszervezési,
tervezési
és
programozási feladatok

mikroszámítógépekre

(Robotron, Sirius, IBM, Alpha-Micro, VT20 A, VT20/IV, Commodore 64, 610, 720)

való elvégzését

MUNKATÁRSAIKAT BETANÍTJUK! RENDELJE MEG NÁLUNK

- **főkönyvi könyvelés,**
- **analitikus anyag-, fogyóeszköz-, terméknyilvántartási,**
- **bér- és munkaügyi rendszerek,**
- **személyzeti nyilvántartás,**
- **állóeszköz-nyilvántartás,**
- **tervezés,**
- **statisztikai feldolgozások,**
- **egyéb célrendszerek**

szervezését, programozását!

A szervezések bevezetését felügyeljük, a programokra egyéves garanciát adunk.
Számítógépek beszerzésében is közreműködünk.

INGYENES SZAKTANÁCSADÁS!

A számítógépek illesztését megtervezzük, elvégezzük.
Személyes konzultációt biztosítunk.

Címünk: 1027 Budapest II., Mártírok útja 24. VIII. em.
Telefon: 151-438
Telex: 22-3124

KOMPLEX SZÁMÍTÁSTECHNIKAI SZOLGÁLTATÁS!

A Ferroglobus szervezési főosztálya nagy adatfeldolgozási hagyományokkal, jól képzett, nagy tapasztalatú munkatársakkal rendelkező számítóközpontja széles skálájú, komplex számítástechnikai szolgáltatásait ajánlja:

— kereskedelmi szakmai, készlet-gazdálkodási, pénzügyi, számviteli, bér- és munkaügyi stb. rendszerek szervezése, programozása és fejlesztése nagy kapacitású, modern ICL típusú számítógépen
— mágneses adathordozóra történő adat-előkészítés — rögzítés
— számítógépidő bérbeadása
— tanácsadás

Ferroglobus

FERROGLOBUS

TEK V.
számítástechnikai főosztály
Budapest VII., Vörösmarty u. 16.
Tel.: 427-338, 202-415

Klubsirató

A sötétben rosszul világított gyárudvaron — a Szellőzőművek gyárában — kóválygo az ember, míg végre rábukkan az ebédlőre. Ott bent igen sűrű a levegő. A délről maradt ételszag masszívan megüli a termet, mégsem válik uralkodóvá, mert mindent elnyom a zsivaj. Aki felnőttként betévedt már egy iskolába, s megtapasztalta a szünetek fület repesztő zaját, annak fogalma lehet arról, hogy micsoda hangzavar uralkodik a Sinclair HCC szekció klubösszejövetelein.

Az egymás mellé állított asztalok soránál látszólag hatalmas a perzsavásár. Villódzó fényeket ontanak a különböző típusú tv-készülékek. A hordozható monitorok, a magnók, ZX81-es, Spectrum mikroszámítógépek kábelek kígyótesteivel vannak összekötve. Mindent elboritanak a vezetékek, a szerteszét heverő magnókazetták sokasága. A szerencsésebbjének jut asztal, oda leteheti a messziről cipelt „készletét". (Mások hiányzó felszerelésüket pótolva papírfecnikre írják programjaikat.) Sokan drukkerként, kíváncsi nézdelőként állnak a magnók, a számítógépek kezelői mögé. Mint a játszótereken, ahol idős szakik verik a blattot, kisebb-nagyobb tétekben téve kockára nyugdijukat, s ahogy velük együtt izgulja végig a lapjárást a semmit sem veszélyeztető kibic. Olyan itt az étteremben is a kép . . . elkeseritően sivár környezet, s a tanácsok szép számmal röpködnek mindenfelé. A nappal étkezde, este klubhelyiség uralkodó zajában innen is, onnan is átszűrődik a jó tanács: ezt a gombot nyomd meg, azt a programot figyeld . . . Ha beválik az ötlet, a jó szándékú figyelmeztetés, ováció fogadja az eredményt.

Meglepően jól megfér egymással a kicsi és a nagy, a diák, a melós és a mérnök úr. Rangkórság itt nem létezik, a közös ügy elnyom minden különbséget. Együtt dolgozik a 12 éves és a tar 60-as, a kérges kezű és a jól vasalt. Nincs is idejük a másik vizsgálgatására. Meredten bámulják a képernyőt, izgatottan nyúlnak a magnó, a számítógép gombjához. Feszülten lesik a képernyőn megjelenő jelzéseket.

A hangzavar és a deprimáló környezet ellenére is az új belépőnek úgy tűnik, e kora esti órában egybegyűlt mintegy 70 ember összeszokott csapatot alkot. Hivessy Ferenc a HCC Sinclair szekciójának titkárhelyettese erről azért másként nyilatkozik. Mint mondja, a társaság meglehetősen változó. A fluktuáció mégsem rajtuk múlik.

Az 1980-ban létrejött HCC klubból néhány megszállott őrült — a sinclairesek — egy esztendő múltán kiváltak, önállósodtak. Kezdetben gépépitéssel foglalkoztak, megtanulták a BASIC nyelvet, ismerkedtek a számítástechnikával. Nyitottak voltak — ehhez máig is tartják magukat —, minden érdeklődőt szívesen, szeretettel fogadtak.

A Tungsram gyár adott otthont, igaz, összesen 4-5 alkalommal. Utána a klubot a TIT vette védőszárnyai alá. Ezzel helyproblémájuk nem oldódott meg. Lakótelepi biciklitárolóban vertek tanyát, s ez a hely be is vált volna, ha az ingyenes klubösszejövetelek, a jó, alkotó légkör nem vonzotta volna úgy a „megszállottakat". Persze, hogy kinőtték a kis helyiséget. Felköltöztek hát a második emeletre. Mivel a lakásban annyian nem fértek el, „kibérelték" a lépcsőházat is. Bizarr jelenetek játszódtak le . . . amikor például elromlott a lift, s a lakók a lépcsőházat is kénytelenek voltak igénybe venni, akkor huzalokon, kábeleken bucskázhattak át a gyanútlan hazatérők . . . Természetesen tartósan ez az állapot nem mehetett tovább. Dönteniük kellett. Feloszlik a csapat, vagy találnak másik megoldást.

A Neumann János Számítógéptudományi Társaságba 1983 óta tartoznak, ám szorító helykérdésük megoldásához — és máshoz sem — nem kaptak semmiféle támogatást.

Hogy mégsem szűntek meg, az Török Zoltán klubtitkár, az alapító szemfülességének és kapcsolatainak köszönhető. A Budapesti Műszaki Egyetem egyik kollégiumába kerültek, két klubhelyiséget is kaptak. Ám a több mint 100 lelkes sinclaires még több számítógépest vonzott, mint korábban. Hamarosan ott sem fértek el . . . Átköltöztek a Kertészeti Egyetem Klubjába, ideig-óráig működhettek is égisze alatt. Aztán megint csak az a bizonyos felduzzadt létszám idézte elő — hiába, nem akartak lemondani a klub nyitottságáról — a távozást. Megint új helyszínen dolgoztak tovább, magukkal hurcolva ismét berendezéseiket. Egy Üllői úti általános iskolába — ismeretséggel jutottak ide is —, ahol egy hétfő este közölték az egybegyűltekkel, hogy kész, vége . . . feleslegessé váltak . . . Aki nem volt akkor ott, az legközelebb csak a társaság hűlt helyével találkozott. Aztán megint a személyes kapcsolat, s a költözködők a Molnár Utcai Ifjúsági Házban dolgoztak tovább. Egymásra hogyan találtak? Az új hely amolyan szájhagyomány útján terjedt el, volt akit elért, volt akit nem . . .

Kár is tovább sorolni a Sinclair klub kálváriáját. Tanulságként levonható, hogy az ingyenes kikapcsolódást nyújtó számítógépes klub nem kell a művelődési házaknak, mert azok inkább a fizető programokat fogadják fedelük alá. Az amatőr, lelkes mikroszámítógépesek pedig „fedezet" híján masináikkal nemhogy a fél várost, de az egészet bejárták már. Közben jutott energiájuk számítástechnikai nyelvekhez tanfolyamokat szervezni, továbbképzéseket tartani, érdekes, jó témákat, megoldásokat egymásnak közreadni. Rendszeresek a programcserék, a tanácsadások, a hardverbővítések, kapcsolási rajzok klubon belüli vándoroltatása. Olyan szakmai folyóiratok cserélnek kölcsönbe gazdát, amelyek nincsenek is közforgalomban, vagy a soványabb pénzűek amúgy nem juthatnának hozzá.

A sehol sem lét, az örökké változó helyszínek azért felőrlik a társaságot. Ne feledjük, a berendezések hurcolása megterhelő erőmutatvány, s ezt nem mindenki bírja . . . Kimaradnak, elkopnak. Kár értük. Elitélni, megitélni mégsem szabad őket. Ezek az emberek szabad idejüket hasznosan kívánják eltölteni — ez dicséretes —, továbbképzésre vágynak az informatika területén — ez becsülendő —, értelmes programokat akarnak maguknak — ez értékelendő. Ha viszont nem megszerezhető csak hosszas kutakodás, csavargás révén, hát érthető szárnyaszegettségük.

Ami azonban elgondolkodtató: miért nem ismerik fel az intézmények, a vállalatok, micsoda jelentős dolog az, hogy az országban, ahol a jelené kellene legyen a számítógép — s talán egyszer a jövőé mégis csak lesz —, az önszerveződő, ingyenes klub keretében egyesek vállalkoznak arra, hogy a számítástechnikai ismereteket továbbadják. Hihetetlen összegekért működnek tanfolyamok — nemritkán 8-10 ezer forintot is elkérnek kurzusonként —, ahová gyakran muszájból kerülnek a hallgatók. Itt meg léteznek önként vállalkozók, szívesen tanulók . . . Az információcserével gazdagodó tudásukat nemcsak önmaguk kedvtelésére fordítanák. Sokan közülük ma már profi számítógépes szakemberek. És akik csak most tanulják e szakmát, azok is hamarosan foglalkoztathatók lesznek a számítástechnika valamelyik ágában. Tanulásukkal nem terhelték az intézményeket . . .

Sajnálatos módon nem ismerik fel kellően a hivatalos helyek, hogy egy-egy jó és tartalmas programokat kínáló klubban olyan szoftverek is íródnak, kerülnek forgalomba, amelyeket esetleg hasznosítani tudnának a gyakorlati életben.

Mi marad addig például a sinclaireseknek? A bizonytalan jelen mellé a kérdőjeles jövő. Meddig és miért lesznek még képesek összejönni, hurcolkodni, újból és újból feltámadni? Lehet, hogy egyszer majd ők sem tudják megadni maguknak a választ . . .

krasznai

C64 modem

Az alább ismertetett megoldást vezeték nélküli rendszeren sok rádióamatőr társammal együtt használom. A C64 mikrogéphez illeszkedő modem egyenáramú jelekből hangfrekvenciás jeleket generál adás irányban, a vett hangfrekvenciás jelekből pedig jelsorozatot állít elő vétel irányban.

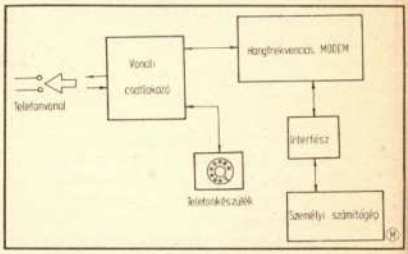
A rendszer felépítése, működése

A rendszer tömbvázlata az *1. ábrán* látható. Adáskor a számítógépből érkező, 1 és Ø bitekből álló jelsorozatok az interfészen keresztül vezérlik a T5 tranzisztort.

Ø bit esetén az OC1 nincs vezérelve, a T5 zárva van, a kollektorában levő 470 ohmos ellenállás a D1 zéner diódát tápfeszültségre húzza. Az IC7a-n keresztül feszültségvezérelt IC6 oszcillátor 2295 Hz-en rezeg. Az IC7b kimenetén ilyen frekvenciájú jel jelenik meg.

1 bit esetén az OC1 fototranzisztora bekapcsol, a D1 zéner nem kap tápfeszültséget. Az IC6 most 2125 Hz-es jeleket állít elő. Adáskor tehát hol az egyik, hol a másik frekvenciájú jelet küldjük a vonalra aszerint, hogy abban a pillanatban 1 vagy Ø bit van-e a 24 vonalas csatlakozó adási pontján.

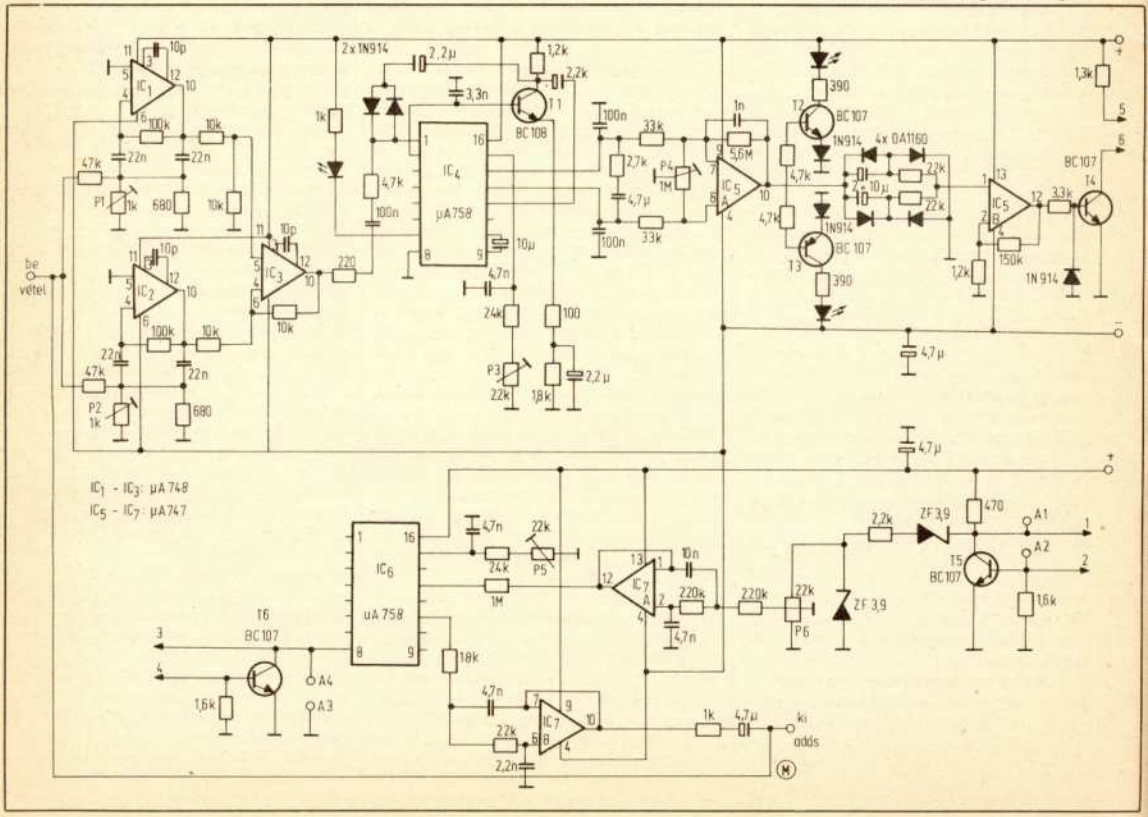
Vételkor a vett hangfrekvenciás jelek szűrőfokozaton keresztül jutnak a demodulátor-egység bemenetére. A szűrőfokozat az átviteli út zaját, áthallásait és egyéb zavaró jeleit szűri ki; ezek befolyásolhatják a demodulátor működését: a PLL (IC4) ugyanis egyéb jelekre is reagál, ami a vételt teljesen megsemmisítheti. A szűrőfokozat beépítésével — az adatátvitel alatt — telefonon is tartható a beszédkapcsolat. Ez megkönnyíti a modemek vizsgálatát és a kísérletezést.

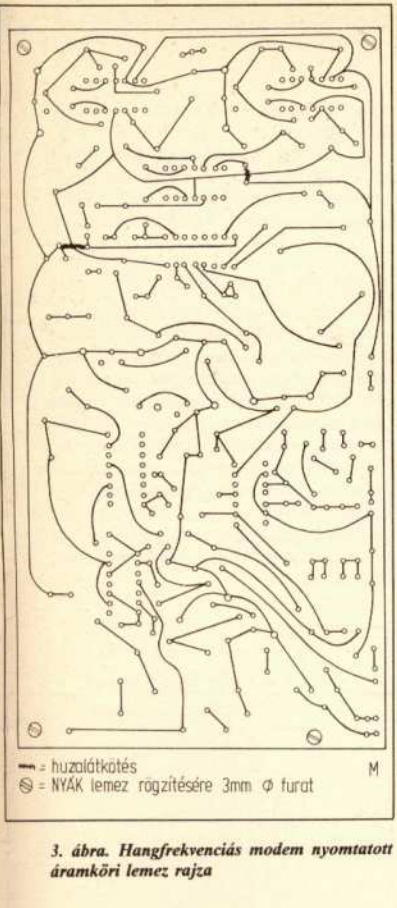


1. ábra. Kis sebességű, telefonvonalon összekapcsolt számítógépes adatállomás felépítésének tömbvázlata

A demodulátor IC4 — μA 758 PLL —

2. ábra. A modem kapcsolási rajza





3. ábra. Hangfrekvenciás modem nyomtatott áramköri lemez rajza

zárt állapotát a 7. lábára kötött LED dióda jelzi. A T2 és T3 tranzisztorokkal vezérelt LED-ek a vett jelsorozat 1 és Ø bitjeit jelzik. A demodulátor-egység IC5b bemenetén négy diódából álló szinthelyreállító áramkör található, amely az ellenállomás adóegységének frekvenciaeltéréséből eredő káros jeltorzulásokat kompenzálja. A vonalról kapott 2125, illetve 2295 Hz frekvenciájú jelsorozatok hatására a demodulátor-egység kimeneti tranzisztora (T4) nyit vagy lezár. Az OC2-n keresztül a vett jelek 1 vagy Ø bitek alakjában jutnak a számítógép 24 vonalas csatlakozójára.

A modem kapcsolási rajzát a *2. ábra* mutatja. A *3. ábrán* a NYÁK-lemez, a *4. ábrán* a NYÁK-lemez beültetési rajza látható. Azokhoz, akik e három rajz alapján vállalkoznak a modem megépítésére, szól a cikk további része.

A modem hangfrekvenciás kimenete és vételi bemenete ugyanarra a vonalra csatlakozik, ezért össze vannak kötve. Ez nem okoz gondot, mert a használt program adás alatt a vett jeleket nem figyeli, vételkor azonban saját adóegységünket ki kell kapcsolni, hogy a vonalról érkező jelek vezéreljék a demodulátort.

A modem üzembe helyezése

A NYÁK-lemez elkészítése után forrasszuk helyükre az alkatrészeket, ellenőrizzük az IC-k, tranzisztorok, LED-ek helyes bekötését, majd egy-egy rövidke vezetékdarabbal forrasszuk össze az A1—A2 és A3—A4 pontokat.

Kapcsoljuk rá a tápfeszültséget a modemre. Kössünk frekvenciamérőt az adóegység kimenetére, és a P5 trimmer potméterrel állítsuk be a frekvenciát 2125 Hz-re. Forrasszuk le a vezetéket az A1—A2 pontokról, és P6-tal állítsuk 2295 Hz-re a frekvenciát. Ezzel befejeztük az adóegység beállítását.

A vevőegység szűrőfokozatát megfelelő frekvenciákra kell hangolni. A vételi bemenetre hangfrekvenciás generátorból 2100 Hz-et adunk —10 dB szinttel. A P1 trimmerrel az IC1 kimenetén (10. láb) maximális jelre állunk. A kimenő jelet oszcilloszkóppal, szintmérő műszerrel figyelhetjük, de ezek hiányában egy magnódeck kivezérlésmérőjét is használhatjuk indikálásra.

A következő lépésben 2320 Hz-et adunk a bemenetre, az IC2 kimenetén (10. láb) P2-vel állítjuk a legnagyobb kimenő jelet — és már kész is van a szűrőfokozat beállítása.

Ezután a frekvenciamérőt az IC4 11. lábára kapcsoljuk.

A P3 trimmerrel 2210 Hz-re hangoljuk a PLL belső oszcillátorát. Ennek beállítását követően az adóegység kimenetét összekötjük a vevőegység bemenetével, mire az IC4 7. lábára kapcsolt LED kigyullad, és a T2 kollektori LED dióda is jelez. Az adóegység A1—A2 pontját összekötve a T3-mal vezérelt LED gyullad ki. A T2 és T3 tranzisztorokkal vezérelt LED-eknek működéskor felváltva, de egyforma fényerővel kell világítaniuk. Ezt a P4 trimmerrel szabályozhatjuk be. A helyes beállításkor közel +2, illetve —2 V mérhető az IC5a 10. lábán.

Ezek után üzemkész a modem.

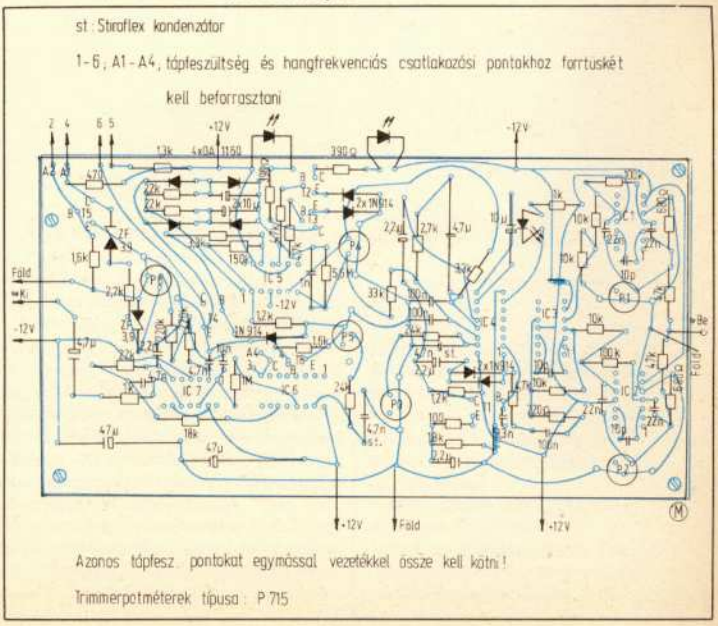
A tápegység

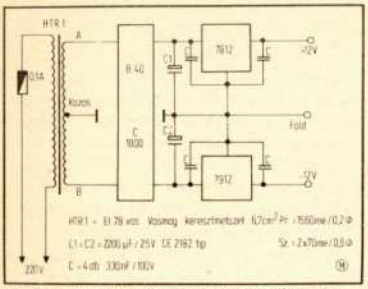
A modem +12 és —12 V-os feszültségekkel működik, amit hálózati tápegységről vagy zsebtelepekről biztosíthatunk. A hálózati tápegység rajza az *5. ábrán* látható.

A T1 hálózati transzformátor 2 × 10 V-os szekunder tekercsei graetz egyenirányítóra kapcsolódnak. Az egyenirányított feszültséget C1 és C2 2200 mikrofarádos 25 V-os elkókkal szűrjük. A +12 V stabilizálását 7812 típusú IC végzi, a —12 V pedig 7912 típusú IC-vel stabilizálható.

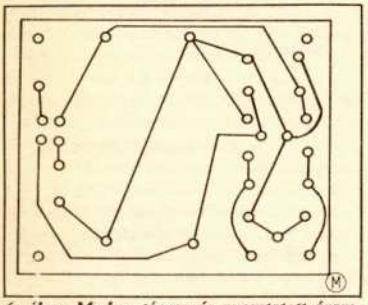
A stabilizátor IC-k TO—220 tokozásúak. A működéskor keletkező hőmennyiség elvezetésére mindkettőhöz egy egy

4. ábra. Modem NYÁK-lemez beültetési rajza

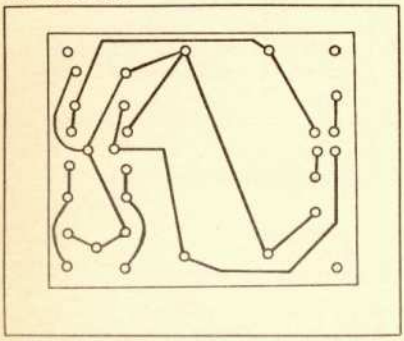




5. ábra. Modem-tápegység kapcsolási rajza



6. ábra. Modem-tápegység nyomtatott áramköri lemez rajza



7. ábra. Modem-tápegység NYÁK-lemez beültetési rajza

50 × 20 × 3 mm-es alulemezt csavaroztam. VIGYÁZAT! A 7912 IC házán — így a hűtőlemezén is — még stabilizálatlan negatív feszültség van. Leföldelni nem szabad!

A *6. ábra* a tápegység NYÁK-lemezét, a *7. ábra* a NYÁK-lemez beültetési rajzát mutatja.

Zsebtelepes tápláláshoz 6 darab telepre van szükség. Ha ezeket összekapcsoljuk, 2 × 13,5 V-ot kapunk. A plusz és mínusz oldali utolsó telepeken egy-egy elem lekötésével állíthatjuk elő a szükséges +12 és −12 V feszültséget.

A személyi számítógép és a modem összekapcsolása

A C64 gépeken a Printtechnik CW/RTTY 84 programot futtattuk. A program a C64 24 vonalas csatlakozójának F pontján adja ki adás irányú jeleit, TTL szinten. A vételi TTL szintű jeleket a csatlakozó J pontjára kell vezetni. Vételkor saját adóegységünk kikapcsolását az E ponton keresztül vezéreljük.

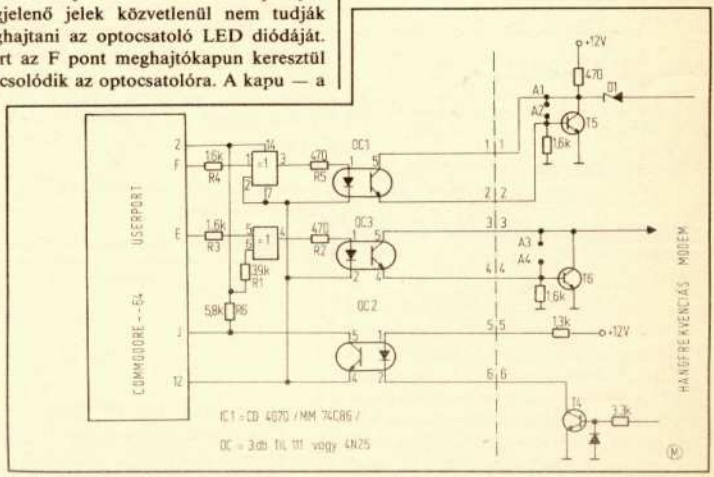
A modem és a számítógép közé egy egyszerű interfészt kell illeszteni, amelynek több feladata is van:

— védi a csatlakozót a túlterheléstől,

— galvanikusan elválasztja a számítógépet a modemtől,

— megoldja a számítógép TTL szintű illesztését a modem 12 V-os tápfeszültségéhez.

Adás irányban a csatlakozó F pontján megjelenő jelek közvetlenül nem tudják meghajtani az optocsatoló LED diódáját. Ezért az F pont meghajtókapun keresztül kapcsolódik az optocsatolóra. A kapu — a kis fogyasztás érdekében — egy CMOS IC egyik kapuja. Az IC a csatlakozó +5 V és föld pontokról kap tápfeszültséget. Az adás irányú optocsatoló az OC1.



8. ábra. Az interfész kapcsolási rajza

Vétel irányban a modem az OC2-t vezérli; annak fototranzisztora továbbítja a csatlakozó J pontjára a vett jeleket.

Az említett programban az adás-vétel átkapcsolás vezérlése a számítógép billentyűzetéről történik: F1 — vétel, F3 — adás. Ez a kimenet a csatlakozó E pontja. Az interfész ezen optocsatolójának meghajtása (OC3) az adásirányhoz hasonló. Az egyetlen különbség, hogy a meghajtókapu itt invertál is. Adóegységünk kikapcsolását a T6 tranzisztor végzi. A T6 nyitásával — adáskor — bekapcsoljuk az IC6-ot, az adási hangfrekvenciás jeleket előállító oszcillátort. Vételkor kikapcsoljuk.

Az interfész kapcsolási rajza a *8. ábrán*, a NYÁK lemezének rajza a *9. ábrán* látható. A *10. ábra* az interfész NYÁK-lemez beültetését szemlélteti.

A rendszer ellenőrzése

A modem működését ajánlatos a másik állomással végzett kísérletek előtt ellenőrizni. A beállított modemet összekapcsoljuk az interfésszel, a számítógépet csatlakoztatjuk a rendszerhez és a modemet magnetofonhoz kapcsoljuk. Adásunkat magnóra vesszük, majd vételre állva visszajátsszuk. Ha rendszerünk kifogástalanul működik, akkor a leadott szöveget visszakapjuk a monitor képernyőjén. Csak ebben az esetben kezdjük meg a kísérleteinket a másik állomással!

A modem telefonvonalra csatlakoztatása

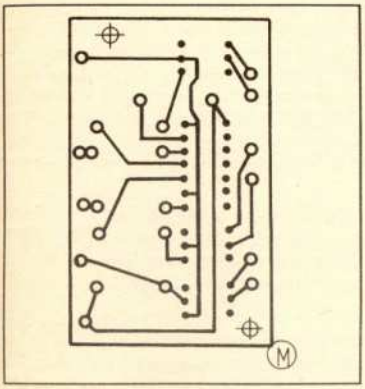
A postai telefonvonalra nem szabad idegen berendezéseket fémesen csatlakoztatni. Ezért a következő megoldást választottam.

Egy telefonkonnektor, egy dugó és néhány méter vezeték felhasználásával „meghosszabbítottam" a telefonvonalat a lakásomban. A hosszabbító vezetékpár egyik ágába sorosan bekötöttem egy 1:1 áttételű hangfrekvenciás transzformátor primer tekercsét, és a hosszabbító végén levő konnektorba bedugtam a telefonkészülék dugóját. Ezzel üzemkész állapotba került a telefonvonal.

A transzformátor másik tekercsére csatlakozik a modem. A transzformátort dobozba szereltem. A modemhez egy kb. másfél méteres árnyékolt vezetékkel, tuchel-dugóval csatlakozom. Ezt a meghosszabbítást és csatlakozási módot természetesen a partnerállomáson is ki kell építeni.

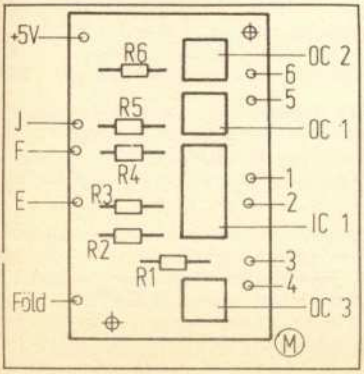
A hangfrekvenciás transzformátor primer oldalával párhuzamosan kötöttem két diódát, melyek az esetleges telefonhíváskor a modembe bejutható káros csengetési feszültség ellen nyújtanak védelmet.

A vonalra csatlakozó rendszer a kísérletezés előtt könnyen kiépíthető, majd egyszerűen megszüntethető. A telefonban saját adásunk vagy az ellenállomásról érkező jelek hallhatók. A rendszer üzemelése alatt beszédkapcsolatot is tarthatunk az ellenállomás kezelőjével. A két állomás között egy időben csak egyirányú átvitel lehet: míg az egyik ad, a másik vesz és fordítva.



9. ábra. Interfész nyomtatott áramköri lemez rajza

10. ábra. Interfész NYÁK-lemez beültetési rajza



Tapasztalataink

A kísérletben részt vevő C64 gépeken az említett programot futtatva dolgoztunk. Telex és ASCII kódokkal, különféle sebességekkel vizsgáltuk az összeköttetést. A legnagyobb kipróbált sebesség 100 baud volt, amelyen üzembiztos és hibátlan volt a vétel. Nagyobb átviteli sebességhez újabb változtatásokra lenne szükség a modemben, de ez nem volt célunk.

Az interfész a 24 vonalas csatlakozó bármelyik — megfelelően programozott — pontjára csatlakoztatható.

Véleményem szerint megfelelő programmal és a sebesség mérséklésével rövidebb programok átvitelét is meg lehetne oldani ezen a módon. Erre vonatkozó tippeket és trükköket jómagam is szívesen olvasnék a Mikroszámítógép Magazinban.

KOVÁCS LÁSZLÓ

Az alábbiakban egyúttal rövid tájékoztatást adunk az alkalmazás előtt szükséges engedélyezési eljárásról, melyet Feczkó Iván, a posta illetékes fejlesztési igazgatóhelyettese bocsátott rendelkezésünkre. (A szerk.)

A távközlési hálózathoz csatlakozó berendezések használatához Magyarországon is — ugyanúgy, mint más országokban — a hálózatot üzemeltető posta engedélye szükséges, a távközlési hálózatot használók (előfizetők) és üzemeltetők érdekében. Azt az állomást (távbeszélő, telex stb.), amelyen nem engedélyezett berendezést használnak, a posta kikapcsolhatja.

Az engedélyt a berendezést gyártó vagy forgalmazó kérheti az engedélyezést megelőző vizsgálatok elvégzésére kijelölt postaszervtől, modemek esetében a Posta Központi Távíró Hivataltól.

A berendezés használatát — megfelelő vizsgálati eredmény esetén — a Magyar Posta Központja engedélyezi. Az értesítést (engedélyt vagy elutasítást) a vizsgálatot végző postaszerv küldi meg az engedélyt kérőnek. A vizsgálat költségeit az engedélyt kérő viseli.

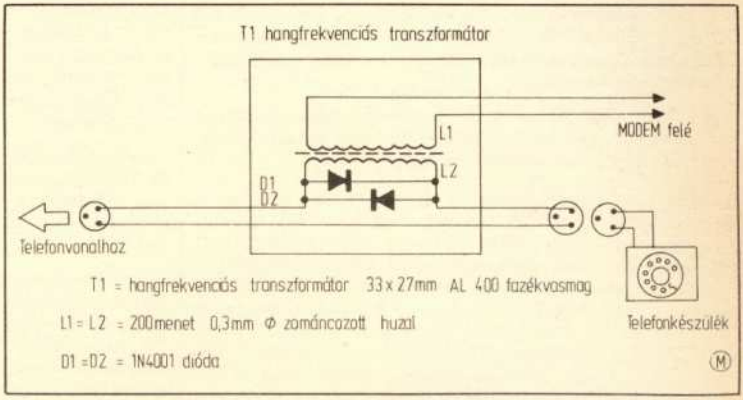
Olvasóink belátására bízva a választást, közlünk egy postai szakvéleményt is Szrnka Györgytől, a modemről. (A szerk.)

A cikkben szereplő műszaki adatokból, valamint az első bekezdésből kitűnik, hogy ez a modem az amatőr rádiózásban használt keskeny sávú 170 Hz frekvenciamodulációt (NBFM) alkalmazza. Ez a modulációs módszer az amatőrök körében a rádiótáviratozás üzemmódban (RTTY) széles körben elterjedt, de csak 50 baudos (illetve az FCC szabványú 45 baudos) modulációs sebességre van előirányozva.

A fenti modulációs módszer és a leírt megoldás távbeszélő-hálózaton való alkalmazása több okból sem mondható szerencsésnek, még ha a postai vizsgálatok alapján egyébként csatlakoztatható lenne is a távbeszélő-hálózatra:

- **Nem felel meg a CCITT (Nemzetközi Távíró és Távbeszélő Tanácsadó Testület) távbeszélő-hálózaton alkalmazandó modemekre vonatkozó egyik ajánlásának sem. A távbeszélő-hálózaton világszerte szinte kizárólag CCITT szerinti modemeket alkalmaznak. A javasolt modem ezekkel nem tud együttműködni.**
- **Rendkívül alacsony az elérhető sebesség.**
- **Csak félduplex átvitelt biztosít.**

A kívánt célra a CCITT V. 21 ajánlása szerinti 300 bit/s-os modem használata látszik célszerűnek.



11. ábra. A telefonvonalhoz történő csatlakozás elvi rajza

OLVASTUNK...

és elgondolkoztunk azon, hogy szeressük-e Canust, Updike novellájának (The Invention of the Horse Collar, megjelent a „Museum and Women" című kötetben, Vintage Books, 1981.) kitalált hősét, aki a történet szerint úgy a X. század elején feltalálta azt a forradalmian új lószerszámot, amely már nem fojtogatta vontatás közben a lovat; az állat teljesítményét háromszorosára-négyszeresére növelte, és ezzel elindította a fejlődést a legsötétebb középkorban.

Canus — a feltaláló — csak a szédületes perspektívát látja, nem érti ikertestvére, Ablatus aggodalmát, és amikor fivére fölgyújtja a szalmából készített, csirizzel összeragasztott „prototípust", megöli őt. Íme egy részlet Updike könyvéből:

„. . . Canus tűnődve ül szalmatetős kunyhójában. Mellette a durván ácsolt és csapolt padon egy halom vázlat, egy sor tompa és éles primitív szerszám, egy ökrökre való iga összehasonlítás céljából, lapockacsont egy lóból, rajta az erőhatások faszénnel felvázolva. Kívül zavartalan sötétség . . . Értelmetlenül elszórva mindenfelé római romok. Egy félig betemetett márvány császár gargantuai fejéből vak szemek meredeznek. Egy vízvezeték kezdődik és szakad meg a semmiben.

Az erdős völgyek útvesztős kolostorokat rejtenek, melyekben csekély értelmű klerikusok silabizálják újra és újra Vergiliust, és azt hiszik róla, valami mágus. A csúcsokra települt, bevágott ablakú kastélyok felhúzzák magukhoz a magasba a falvakat, mint hölgyek szoknyáikat, mikor pocsolyán kelnek át. A viszonylagos rend e szigetei között torokhangú törzsfőnökök — még orgyilkosok inkább, mint lovagok — száguldanak fel és alá, orditásuk inkább rontott latin még, mint francia, letapossák a kényes barázdákat. Canus egyike azoknak, akik ezt a bizonytalan földet művelik. Itt nógatja eke elég fogott, fulladozó, tántorgó lovát. Nem tetszik neki a dolog, bosszankodik. Kell, hogy legyen valami más megoldás, valami jobb. És íme, most kezei között van az első, szalmából és csirizből összetákolt nyakló!

A kunyhó ajtaja kivágódik. Belép Ablatus, Canus ikertestvére . . . Ez mi? — kérdezi olyan nyelven, melynek archaikus dallama fülünk számára mindörökre elveszett.

Canus lelkesen magyarázni kezdi találmányát. Egy új világot ír le: a vonóerő négyszeresére növelését, megjavított mélyszántást, gyorsabb szállítást, egy jobban szervezett, jobban táplált kereszténységet, katedrálisok emelkedését, a vikingek és mohamedánok kiűzését. Kereszteshadjáratokat lehet majd inditani, kamatra lehet majd pénzt kölcsönözni, kialakulhat egy középső osztály — és mindez a sok áldás ebből a durva, összecsirizelt szalmaabroncsból származik majd. A végleges nyakló természetesen kipárnázva, bőrből fog készülni majd, egyre tökéletesebb kivitelben, hogy ne dörzsölje fel a ló nyakát, és a ló számára az erőkifejtést minél kellemesebbé tegye.

Ikertestvére hosszú előadásának vége felé Ablatus alig tudja visszafojtani indulatait. Sarokba vágja sarlóját, melynek alakja alig változott azóta, mióta az egyiptomiak először arattak vele. Fakó szeme szikrákat szór: saját testvérem — nyögi ki végül — maga az ördög!

Nem, mondja Canus, és meglepetésében feláll a padról. Angyal inkább, aki mind az állatokon, mind az embereken segít egy ügyesebb, jobb hatású — és itt önkéntelenül a latin mechina-hoz nyúl vissza megbotló nyelvvel, és ezzel új szót alkot — *masinával*. Létrehoztam — és itt most valójában új szót kell alkotnia — a *lóerőt*.

De te ezzel véget vetsz az anarchiának, mely mindnyájunkat felszabadított! Ablatus aggódó víziójában üzötten méri lépteivel fel és alá a kunyhó poros négyszögét. Visszahozod ránk Róma nyomasztó rendjét, mely császárokat őrjített meg mielőtt még a rabszolgák tengere kiapadt volna, még mielőtt a Birodalom — mint egy zsák kő — elsüllyedt volna . . . *Oh Canus frater meius*, gondolkozzál! A püspökök tehetetlenségével és a barbárokkal Isten ezer részre törte Rómát, hogy az emberek fellélegezhessenek . . ."

És a végén Ablatus felgyújtja a prototípust, mely nem semmisül teljesen meg, Canus viszont egy határozott mozdulattal agyonvágja, majd szép nyugodtan eltemeti ikertestvérét, a jövőt a *haladástól* féltő Ablatust.

„A harangszó a közeli kolostorból — mint ezüstszál, melyet átvetettek a szakadék felett — idáig húzódik a homályos völgyből. A sötét középkor lassan véget ér. Ahogy Canus ásójára támaszkodik, arcát megsímogatja az esti szellő — és erről eszébe ötlik: ebben is rejtőzködik erő, amit ki lehetne használni. És vitorlákat képzel el, és fogaskerekeket és tengelyeket. Kunyhójában alváshoz készülődik. Izmai jólesően fájnak az elvégzett hasznos munkától . . . holnap, gondolja, felvázolja majd a vízszintes tengelyen forgó, szélhajtotta malmot is és . . . holnapután — *Deus volens* — . . . a talicskát."

Eddig Updike szövege az idézőjelek között. Reméljük, ezt a kötetet is lehet majd előbb-utóbb a mostaninál avatottabb és pontosabb fordításban — és főleg teljes terjedelemben, nemcsak illusztrációként kiragadott idézetek formájában — magyarul is olvasni. Megérné!

De térjünk vissza a kiinduló kérdéshez: szeressük-e Canust, a feltalálót, akit szemmel láthatólag nem érdekelnek a technikai fejlődés veszélyei. Aki képtelen másképpen viselkedni: amit meg *lehet* jobban csinálni, azt meg *kell* jobban csinálni. Nemcsak azért, mert ha „én nem csinálom meg, megcsinálja más", hanem, mert egyszerűen kell. Ez a küldetése. Talán az sem érdekelné, ha tudná, hogy egy későbbi korban olyan szépen fogják írni: „Nincs szabadjegy, jól haladni a korral . . ." Ha megzavarják, ha tönkreteszik a munkáját, szép akkurátusan agyonüti azt, aki megzavarta, majd el is temeti, de esze közben már messze, újabb *innováción* jár. Egyoldalúan csak az áldásokat látja, a csapdákat, a veszélyeket nem.

Szerintem ne szeressük, de ismerjük fel és viseljük el. Ha ilyen, gondolkozzunk helyette is! Ne próbáljuk szétrombolni világát. Tiszteljük azt, amit csinál. Főleg: *értsük meg*, mit akar. Ha egyoldalú, egészítsük ki mi. Olyasmit tud, amiben mi általában nem nagyon jeleskedünk.

Az ideális megoldás persze a két ikertestvér — Canus és Ablatus — egyesítése lenne. Olyan valaki, aki egyszerre lenne zseniális feltaláló, de a társadalmi csapdákra, veszélyekre érzékeny lélek is. Ezt az innováció mindkét oldalára egyaránt figyelő, ideális újítót lehetne igazán szeretni. Rajtunk is múlik, hogy több legyen belőlük.

— KE —

µINFORM

A tartalomleírások az alábbi folyóiratokban megjelent programlistákról készültek:

A folyóirat neve	Kódja
64'er Magazin	64er
Chip Magazin	chip
Commodore Microcomputers	comi
Compute!	cute
Happy Computer	happ
Run /USA/	run
Run /NSZK/	run2
ZX Computing Monthly	ZXCM

A tartalomleíró szóláncokat permutáltuk, a szóláncvariánsokat pedig alfabetikusan rendeztük.

A tartalomleírás egy szóláncból áll, majd a listában ezt követi a forrás megjelölése a folyóirat azonosítójával, a megjelenés dátumával és a cikk előkeresésékez a kezdő oldalszám és a terjedelem megadásával. A mellékelt lista értelmezéséhez még az alábbiakat kell tudni. A tartalomleírás szóláncában elsőként a téma átfogó megnevezése, utána a számítógéptípus(ok), ezt követően a szűkebben jelölt tartalom meghatározása szerepel, majd esetlegesen néhány, a közleményt minősítő adat (például : cikksorozat).

A forráshely karaktersorozatát nyíl vezeti be, melyet a / jelig a folyóiratok azonosítója, a két / jel között az évszám, folyóiratszám és kötőjellel a kezdő oldalszám követi, a végén pedig a közlemény teljes oldalterjedelme áll.

A folyóiratok a SZÁMALK szakkönyvtáraban (Budapest,XI.,Szakasits Á. út 68. Nyitva: 8-tól fél 5-ig. Tel.: 853-111/251) is föllelhetők. A kiválasztott anyagról másolat rendelhető az alábbi formában:

SZÁMALK Szakkönyvtára
Budapest, 112. Pf.: 146. 1502

Megrendelem a Mikroszámítógép Magazin 1987/ sz. alapján a következő folyóirat-oldal-másolatokat:

Kód: ______ Példányszám: ______
Kód: ______ Példányszám: ______
Kód: ______ Példányszám: ______

A megrendeléshez csatolom az oldalankénti 8,- Ft-os szolgáltatási díj befizetését igazoló csekkszelvényt.
Dátum, név, pontos cím.

```
    star nx-15■commodore 64■ibm Pc-kompa
    tibilis/a4/a3,18x27 tu■2495 dm
    ->run2/87.01-41/1
PROGRAMLISTA
    adatatvitel■commodore 64;128■(modem
    )■fajltovabbitas ->cute/87.01-75/3
PROGRAMLISTA
    adatbazis■commodore 64■lemezegyseg(1
    541)■relativ fajlok kezelese
    ->run2/87.01-18/8
PROGRAMLISTA
    apple ii■nyomtatas■(Printer master)■
    n ->cute/87.01-67/4
PROGRAMLISTA
    commodore 128■(reminder 128)■határid
    o nyilvantartas ->run/87.01-49/5
PROGRAMLISTA
    commodore 128■grafika■(fromiPaint)■b
    otkormannyal vezerelheto menu
    ->run2/87.01-99/3
PROGRAMLISTA
    commodore 16■(directory-sorter)■leme
    ztar karbantartas
    ->run2/87.01-126/3
PROGRAMLISTA
    commodore 16■jatek■(space fort)
    ->run2/87.01-116/4
PROGRAMLISTA
    commodore 16;116;Plus/4■egylepes-szi
    mulator ->run2/87.01-113/3
PROGRAMLISTA
    commodore 16;116;Plus/4■utmutato a c
    olor-utasitas hasznalatahoz
    ->run2/87.01-90/2
PROGRAMLISTA
    commodore 16;Plus/4■grafika■Programo
    zasi tippek ->64er/87.01-79/3
PROGRAMLISTA
    commodore 64■(crossi)■goto/gosub ela
    gazasok ellenorzese
    ->run2/87.01-128/5
PROGRAMLISTA
    commodore 64■(fast key)■egybillentyu
    s utasitasok eloallitasa
    ->cute/87.01-79/2
PROGRAMLISTA
    commodore 64■(speed-sort support)■le
    meztar-kezeles/rendezes
    ->run2/87.01-102/11
PROGRAMLISTA
    commodore 64■(unsplat)■visszamentes
    lezaratlan fajlokbol
    ->cute/87.01-83/2
PROGRAMLISTA
    commodore 64■giga-cad■fajltomorites■
    betolto rutin sajat Programokhoz
    ->64er/87.01-87/4
PROGRAMLISTA
    commodore 64■grafika■(testbildgenera
    tor) ->64er/87.01-61/3
PROGRAMLISTA
    commodore 64■grafika■hypra-basic■bit
    terkep modulok ->64er/87.01-83/5
PROGRAMLISTA
    commodore 64■grafika■nyomtatas■nagyi
    tas falitapeta-meretre
    ->64er/87.01-64/2
PROGRAMLISTA
    commodore 64■grafika■nyomtatas■nyolc
    szoros nagyitas poszterkeszitesehez■m
    ps801 802 ->run2/87.01-124/2
PROGRAMLISTA
    commodore 64■jatek■(word wars)■keres
    ztrejtvenyszeru szokirakas
    ->run/87.01-74/5
PROGRAMLISTA
    commodore 64■jatek■damajatek
    ->64er/87.01-52/4
PROGRAMLISTA
    commodore 64■jatekProgram keszites■k
    ep/hanghatas keltese■rutingyujtemeny
    ->run2/87.01-96/3
PROGRAMLISTA
    commodore 64■naptarkeszites
    ->run/87.01-89/2
PROGRAMLISTA
    commodore 64■Pozicionalt kiiratas■Pr
    int at-utasitas beepitese
    ->run2/87.01-95/1
PROGRAMLISTA
    commodore 64■rendezo rutinok -
    ->run/87.01-13/2
PROGRAMLISTA
    ram-floppy■tarolobovites■bovito modu
    l■commodore 64■vezerloProgram a 128k
    -hoz ->run2/87.01-86/4
PROGRAMLISTA
    sinclair spectrum■(window)■felhaszna
    lo altal definialhato csatornak
    ->zxcm/87.01-66/4
PROGRAMLISTA
    sinclair spectrum■discovery■lemezegy
    seg■rutingyujtemeny
    ->zxcm/87.01-70/4
PROGRAMLISTA
    sinclair spectrum■grafika■muszaki ra
    jz■vonalrajzolas ->zxcm/87.01-96/5
PROGRAMLISTA
    sinclair spectrum■jatek■(the war of
    the shires) ->zxcm/87.01-75/4
PROGRAMLISTA
    sinclair spectrum■jatekProgram keszi
    tes■karaktertervezes■szinek es diszi
    toelemek hasznalata
    ->zxcm/87.01-18/4
PROGRAMLISTA
    sinclair spectrum■jatekProgram keszi
    tes■terkepszeru hatter eloallitasa
    ->zxcm/87.01-32/5
PROGRAMLISTA
    sinclair spectrum■rutingyujtemeny
    ->zxcm/87.01-51/3
PROGRAMLISTA
    sinclair spectrum■sprite■jatekProgra
    m keszites■(deresolution)■beepitheto
     rutin ->zxcm/87.01-42/3
PROGRAMLISTA
    sinclair spectrum■szovegfeldolgozas■
    (specword 48/128) ->zxcm/87.01-28/3
PROGRAMLISTA
    sinclair spectrum■szovegkezeles■(tex
    t miser) ->zxcm/87.01-49/2
PROGRAMLISTA
    sinclair spectrum■zene■(Patch sequen
    cer)■vezerles modositas
    ->zxcm/87.01-37/1
PROGRAMLISTA
    sinclair spectrum 91■zene■klaviatura
     modositas ->zxcm/87.01-12/5
PROGRAMLISTA
    sprite■commodore 64■jatekProgram kes
    zites■assembler PeldaProgram
    ->run2/87.01-92/3
PROGRAMLISTA
    szovegkezeles■commodore 64■nyomtatas
    ■szoatvitel sorvegeken
    ->run/87.01-94/3
PROGRAMLISTA
    zene■commodore 64■(music maker 64)
    ->cute/87.01-62/3
```

A Tejút lassan távolodott. Már csak annyi látszott belőle, mint szellős nyári éjszakán egy tóparti kisvárosból, ha a tó közepéről figyeljük. Csak a legvakítóbb csillagok fénye érte utol a karaván.

Nem mindennapos menet távolodott a galaxistól. Elöl egy kocka repült, mögötte palástként szétterülve hét műholdja követte. A kocka falait bárki betonnak nézhette volna, de a műanyag, amiből készült, összehasonlíthatatlanul ellenállóbb volt. Ez az anyag bármilyen belécsapódó tárgy energiáját szétvezette, így védve a Gépet, amelyet rejtett és táplált. Táplált, mert egyben energiaelnyelő rendszer is volt.

A Gép szinte teljesen kitöltötte a kockát. Csupán az egykori szerelőfolyosókon lehetett volna közlekedni, de már nagyon régen nem járt itt senki.

Állandóan ellenőrizte műholdjai helyzetét, vigyázott, nehogy túlságosan megközelítsék. A kockában teljes volt a nyugalom. A karbantartó robotok energia nélkül, tetszhalottként gubbasztottak a helyükön, a falhoz erősítve. A katasztrófa négy műholdját tette tönkre. Nagyon gyakran siklott végig ezen az eseményláncon.

Új tárolóterületekre volt szüksége. Szelektálni kezdte a múltat, tárolóiból törölnie kellett a használhatatlan, felesleges információkat.

Elsőként a dátumokat törölte. Az emberek mindenféle kitalált eseményekhez kötött időszámítását feleslegesnek ítélte. Módjában állt, hiszen mestere — egy szakállas férfi, aki önmagát nevezte így — saját időszámítással látta el, belétáplálta a gépi nulla évet. Ekkor fejezték be az építését. Miután befejezte ezt a műveletet, már minden esemény csak a saját időskáláján volt elhelyezve, de már ez sem sértett senkit.

A Gép vitte az emberiség utolsó fél évezredének pontos történetét és az addig felhalmozott összes ismeretanyagot.

A lift már-már a szabadesést megközelítő gyorsasággal száguldott a mélybe. A kabinban gravitációkiegyenlítő vigyázta az utasok jó közérzetét, unalmas álldogálássá változtatva a zuhanást.

Most mindössze egyetlen személy tartózkodott a liftben. Alacsony férfi volt, automatikus mozdulatokkal tolta vissza a szemébe hulló fekete hajtincset, majd beletúrt szakállába. Szemei mélyen beágyazódtak az arcába, százhatvan centije, görbe háta, csapott vállai nem sejtették, hogy egy rendszer ura, egy információs rendszeré. Gyakran mondogatta munkatársainak, hogy ő tulajdonképpen egy temetkezési vállalkozó: semmi köze az élethez, de a műve alapján ítélik meg majd a halottat.

George Keden rendszerprogramozó. Övé volt a rendszer, ő alkotta. Tanító volt, és hogy hogyan születik a gyerek, az egy csöppet sem izgatta. A programokat vagy ő írta, vagy Chaplin-ábrák formájában kiadta segédeinek, akiknek ezek alapján csak kódolniuk kellett a programsorokat. De még így sem bízott meg bennük, órákig böngészte az elkészült programrészeket, hogy tud-e valamit finomítani vagy egyszerűsíteni rajtuk.

Utoljára jött le ide, és ez a tudat rettenetesen elkeseritette George Kedent. Életének utóbbi éveit teljesen kitöltötte a Gép.

A lift halk csengetéssel jelezte, hogy megérkezett. Az ajtó kitárult, és Keden kilépett a barna padlószőnyeggel borított folyosóra. Innen nyiltak a szobák, amelyeket — ő legalábbis — második otthonának érzett, de vége. Reggel felviszik a még itt lévő dolgaikat a felszínre, és vége.

A folyosóra szűrődő zajokból tudta, a búcsúztatás már megkezdődött. Társai a munkának mondtak viszlátot, de ő . . . Ő valami teljesen mástól búcsúzott, amit meg sem tudott fogalmazni rendesen.

— Hiszen csak egy gép — mondta magának. Végigment a folyosón, és benyitott az ünneplők szobájába.

Izmos kezek csapkodták a hátát, és valaki egy műanyag poharat nyomott a kezébe. Ezek az emberek boldogok voltak, nem úgy, mint ő. Hallotta, hogy az egybegyűltek egy emberként követelik tőle a pohárköszöntőt. Mosolyogva bólogatott, mire a neonok percegésén kívül minden elhallgatott. Várták a beszédet, így kénytelen volt belekezdeni.

— Elkészült — kezdte rekedt hangon, ivott egy kortyot, majd folytatta — elkészült a Gép, és ezzel kicsit üresebb lett az életünk! Mindannyiunké. Megszűnt a cél, azaz elértük a célt, amiért éveken keresztül dolgoztunk. Megalkottuk, vagyis megépítettük az emberi emlékezet, az emberi kultúra teljes tárát. Ez a Gép talán örökre az emberi kultúra hirdetőjévé válik. Fémteste őrzi mindazt, amit az ember valaha létrehozott és átélt. Ismer mindent, ami jellemezheti az embert. Érti az összes élő és holt nyelveket. De ti is jól tudjátok, hogy ha csak ennyi lenne az egész, akkor egy könyv és egy magnó bőven megtenné helyette, de műholdláncán folyamatosan bővíti tárolói tartalmát, és összekapcsolja ezeket. És még valami, amire egyetlen más gép sem képes! Majdnem ugyanúgy következtet, mint egy ember. Hála Mr. Dicksonnak, akit joggal nevezhetünk a mesterséges intelligencia első pszichológusának, az elv, ahogy asszociál, ahogy paránvi összefüggések alapján elvonatkoztat, az szinte megegyezik egy átlagember gondolkodásával. Ez a Gép időtlen. Amíg lesz egy gyufányi fény, addig működik, és ha majd teljes sötétség veszi körül, akkor is újraéled az első fénycseppnél!

Ez a Gép olyan egy kicsit, mint mi, hiszen mi építettük, talán mondhatom, hogy a tudatunk egy része! Cheerio! — fejezte be pohárköszöntőjét George Keden, és szájához emelte a poharát.

A hangulat egyre emelkedettebbé vált. Keden képtelen volt együtt örülni a többiekkel. Éjfél felé nem bírta tovább, felállt és kiment a szobából. Szótlanul elballgott a liftig. Belépett, és megnyomta a legalsó gombot.

A lift most sokkal lassabban haladt. Halk csilingelés jelezte, hogy megérkezett. Kilépve személyazonosító kártyáját az ellenőrzőegységbe helyezte.

Az ajtó kitárult, és ő egy harminc méter élű kockában találta magát. A mérnöki panel fényei vadul villogtak, jelezve, hogy a Gép dolgozik, gyűjti az információkat.

A vezérlőegységhez ment, maga elé húzta a terminálját, majd a parancscsatornán belépett a rendszerbe.

A képernyőn szédületes iramban villogtak a kommunikációs rendszer tesztjei. Keden kitűrt a szeméből egy hajtincset, és végigsimította a szakállát. A képernyő elsötétült, majd egyetlen felirat jelent meg:

„Kérem az azonosítóját".

— Mester — ütötte be Keden, pedig nem akart dolgozni.

„Mester" — ismételte meg a Gép. — „Azonosító elfogadva, kommunikálásra kész".

— Beszédrendszer — utasította Keden a Gépet.

„Beszédrendszer" — jelent meg a képernyőn, majd egy gépi hang szólalt meg recsegve — kommunikálásra kész!

Keden szórakozottan játszott a billentyűzeten. — Nem akarok most dolgozni — kezdte —, csak körül akartam nézni még egyszer! — Kicsit várt, hátha válaszol a Gép, de nem tettek fel neki kérdést, ezért hallgatott. — Az emberek gyermekeket szülnek, és megpróbálják olyanná nevelni őket, amilyenek ők maguk szerettek volna lenni. A gyerekeikben látják önmaguk meghosszabított életét. Ők tévednek! De nekem te vagy! Én benned élek tovább! Bár még legalább egy év kellene, hogy befejezzelek, de nincs pénz. Nagy veszély ez, nagyon nagy. Soha nem tudod majd eldönteni, hogy mi az, ami értelmetlen. Ez még hiányzik belőled, de fogalmam sincs, hogy miképp lehetne ezt kiküszöbölni. Neked van valami ötleted?

— Törölni kell! — válaszolt a gépi hang.

— Törölni! Nem, néha az értelmetlennek tűnő dolgok mozgatják az értelmet is! Hol a határ? Ezt kellene megtalálni! Mi választja el a kettőt? De nem azért jöttem, hogy itt gondolkozzam, hanem búcsúzni. Remélem, mindig itt állsz majd! — Elhallgatott, majd kisvártatva ismét megszólalt: — A Mester ma meghalt! — Kihúzta a terminál interfészét, és kigördítette a képernyőt a kockából. Az ajtó hangos szisszenéssel zárult be mögötte.

Ez pontosan 3-ban történt, gépi időszámítás szerint.

A Mester 3-ban meghalt, George Keden 17-ben. Ezt a Gép egy műholdjelentésből szűrte ki.

Most itt repül a világűrben. Itt volt minden, ami a földi kultúrából megmaradt. A Gép szelektált, válogatott az információk között, mi az, amit töröljön és mi az, amit megőrizzen. Úgy döntött, ezt az eseményláncot megtartja, pontosan így, ahogy összesűrítette.

A Gép leállította a csillagászati adatok gyűjtését. Tovább válogatott a múlt eseményei között.

Eleinte törődtek vele, ő volt a legnagyobb adatbázissal rendelkező számítógép. Kétezernél több terminálján keresztül bármely, a múltra vagy jelenre vonatkozó információt lekérdezhettek tőle. De ahogy folytak az évek, a kérdések egyre ritkábban követték egymást. Egyre több kihelyezett terminálját szüntették meg. A képernyők, a billentyűzetek a helyükön porosodtak, csupán egyetlen apró panelt szereltek ki, a műholdkövető panelt. Enélkül elérhetetlenné vált. Ez a panel nemcsak kommunikációs sugarak irányítására volt alkalmas, de szinte bármely könnyű fegyver célkövető egységében használhatták. Tervezői drót nélküli adattovábbítással álmodták meg a Gépet. A lekérdezés terminál—műhold—Gép vonalon valósult volna meg. Gépi időszámítás szerint 494-re mindössze pár tucat terminálja volt már csak működőképes, de azokat sem használták sűrűn.

A műholdak hírei is megritkultak és egysíkúakká váltak. A Konföderáció, ami egyesítette az egykori országokat, felbomlóban volt. Különböző érdekcsoportok ismét darabjaira akarták szabdalni a Földet. Előkerültek a „régen megsemmisített" atomfegyverek, de csak kiegészítő szerep jutott nekik az új, bolygók megsemmisítésére is alkalmas pusztító eszközök mellett.

A Mester sejtése igaznak bizonyult. A háború kitört, és napokig tartott csak. A geológiai mérések iszonyatos katasztrófát vetítettek előre. A Gép a Földtől távolabbi pályára irányította műholdjait. Még egy utolsó képet kapott a hófehér henger alakú házról és a parabolaantennákról, amelyek 300 méterrel felette, a Föld színén biztosították működését.

A fegyverek pusztító erejét nem volt képes elviselni a bolygó, és felrobbant. A kocka pörögve vágódott ki az űrbe. Ügyelt arra, hogy műholdjai kikerüljék a Föld szerterepülő darabjait. A saját pályáját nem számította ki, ezért pörögve vágódott négy összelánccolt műholdjába. Pörgése lelassult, de a négy műhold használhatatlanná vált az ütközésnél.

Amikor kikerültek a veszélyzónából, a Gép ellenőrizte a megmaradt műholdakat. Ezeket a szerkezeteket kisebb pályamódosításokhoz elegendő üzemanyaggal látták el, és ahhoz, hogy követni tudják a kockát, majdnem a teljes készletük felét el kellett használniuk. A Gépnek szüksége volt rájuk. Nélkülük vak és süket lett volna. A kockában elhelyezett érzékelők csak egy kb. kétszáz méteres körzetből tudták összekaparni az adatokat.

Hétszáz gépi év folyt el halkan, és a Gépet ismét riadóztatták. Ez most komoly volt, a lehető legkomolyabb. A vészjelzések szakadatlanul érkeztek az összes műholdról. A Gép küldetése még el sem kezdődött, és a pusztulás most nagyon közel járt hozzá.

Sikerült azonosítania a veszélyt. Egy gömb közeledett felé szédítő sebességgel. Maga mellé irányította a műholdakat, közvetlenül a kocka falai mellé. Kiszámította a kitérési pályát, majd a műholdak hajtóműveit egyetlen motornak tekintve, letért a gömb pályájáról.

Az idegen valami elsuhant mellette. A Gép visszaállította a megszokott palástelrendezést. A vészjelzések abbamaradtak. Ellenőrizte a műholdak üzemanyagszintjét. Minimális volt, de egy esetleges bolygó körüli pályára állításhoz még elegendőnek találta.

Nem tudta, mi száguldott el mellette. Lehívta a találkozás adatait. A gömb átmérője 6762 km volt.

— Majdnem akkora, mint a Mars — állapította meg.

Folytatta a képek elemzését. A gömb tulajdonképpen egy láthatatlan burok volt, amit a benne repkedő szikladarabok jeleztek. A meteoritszerű repülő dolgok sohasem hagyták el a gömböt.

Erőtér — talált rá a magyarázatra. Egy bolygót körülfogó erőtér, ami nem enged senkit és semmit se ki, se be.

Vajon milyen fejlettnek kell lennie egy civilizációnak, hogy ne pusztítsa el önmagát? Az embernek még nem volt ennyire fejlett technikája, mégis sikerült elpusztítania önmagát.

Nagyon jól tudta, hogy a saját maga által feltett kérdésekre nem fog tudni válaszolni, ha nem képes újabb információkat szerezni.

A csillagok már körbefogták. Besodródott az új galaxis „előszobájába". Kiszámította, hogy melyik bolygóba csapódik be. Összehasonlította a kocka ellenállási adatait a becsapódásnál keletkező rombolóerővel. A számításai szerint egyben kell leérnie. Kiszámította a becsapódás pontos idejét, rögzítette az összes olyan elemét, ami a becsapódásnál elmozdulhat. Minimális energiára kapcsolt és várt.

Mikor elérte a kellő távolságot, bolygó körüli pályára állította műholdjait.

Még tíz perc, és eléri a bolygó gázburkát. Megállíthatatlanul zuhant az új világ felé. Hangérzékelői folyamatosan közvetítették a bolygót körülvevő gázban való súrlódás hangjait.

A becsapódást viszonylag simán megúszta. Elméleti számításai beigazolódtak, a kocka falai szétvezették a becsapódás energiáját. Mindenre kiterjedő általános ellenőrzést hajtott végre.

Adatokat kért a műholdaktól, hogy meghatározza tényleges helyzetét. Energiával látta el a karbantartó robotokat, és kinyitotta az ajtót. Jelzést kapott, hogy a kocka bejárata a talaj szintje alá került, mert csúcsával fúródott a bolygó felszínébe.

A műholdak jelentették, hogy egy lény közeledik felé. Ezt hamarosan közvetlen érzékelői is megerősítették. A lény körbejárta, majd távolodni kezdett tőle. Végigfényképeztette a bolygót, de csupán egy települést észlelt.

Az ellenőrző tesztek végrehajtása után egyfajta bizonytalanság tört rá. A főrutin és az alprogramok mintha kicsit eltértek volna egymástól. Újra letesztelte magát, és rátalált egy eddig használaton kívüli programcsomagra.

Tudta, hogy van fegyvere.

A játékfa és kiértékelése 5.

Az eddig ismertetett faelőrerendezési eljárások alkalmazásával megbizonyosodhattunk arról, hogy velük jelentősen felgyorsul a játékfa elemzése, és ezért nő a program játékereje. A megtakarított idő lehetővé teszi, hogy az algoritmus egy-két féllépéssel mélyebben értékelje az állást, vagyis mélyebben elemezze a változatokat. Így a horizonteffektus csak később jelentkezik, és a számítás is sokkal pontosabb lesz. Ezért a — többségében heurisztikán alapuló — előrerendezési eljárások kidolgozása a sakkprogramozás egyik sarkalatos pontja; nagyon sokan próbálkoznak újabb és újabb eljárások kidolgozásával.

A programozók azt várják, hogy az előrerendezés során a különböző heurisztikák kombinálásával egyre jobban közelítsenek a fa valóságos rendezéséhez. A kanadai Alberta Egyetem tanára, Jonathan

Schaeffer kidolgozott egy eljárást, amely mérései szerint hatékonyabb az eddigieknél. Elméletének a *történeti heurisztika* nevet adta.

Ez a heurisztikus eljárás a gyilkos lépés elméletéhez hasonló módon a játékfa már értékelt részét veszi alapul a fa többi részének előrerendezéséhez. Ez úgy történik, hogy minden egyes csomópontban jegyzi az algoritmus, hogy a belőle kiinduló lépések hányszor bizonyultak eddig a legjobbnak. A következő faág vizsgálatánál az előző adatokhoz ez utóbbit is hozzáadja, és az így kapott eredmény szerint rendezi a lépéslistát közvetlenül az új lépésgenerálás után. Tehát a legtöbbször legjobbnak bizonyuló lépés kerül előre, majd a második, harmadik, negyedik legjobbnak bizonyuló stb. Az eddig talált legjobb lépés jegyzésénél Schaeffer szerint figyelmen kívül lehet hagyni a keresési mélységet, és elég csak egy táblázatot használni a lépések regisztrálására. Ezt azzal indokolta, hogy mindig a végcsomópontban van a legtöbb lépés, és csupán őt — esetleg hat — féllépéses kutatás esetén a fa belső része elhanyagolható. Ez viszont már nem teljesül a hét vagy ennél több féllépésű kutatás esetén, mert a programok többsége csak ritkán „merészkedik" ilyen mélységekbe. Ilyenkor az előző számainkban már ismertetett hagyományos technikák a megfelelőek.

A történeti heurisztika gyakorlati alkalmazásához legcélszerűbb egy 64×64-es mátrixot deklarálni. Így biztosak lehetünk benne, hogy minden lehetséges lépéshez tartozik a mátrixnak egy egyértelműen meghatározott, könnyen kiszámítható indexű eleme. A keresési eljárás meghívása előtt minden alkalommal nullázni kell a mátrix elemeit, majd a kutatás során minden kiértékelt csomópont után az adott csomópont legjobb lépésének megfelelő mátrixelemet növelni kell eggyel. A lépésgenerálás alkalmával ebből a mátrixból kell kiolvasni a lépésekhez tartozó pontértékeket, és eszerint kell azokat rendezni. Ez nagyon jó közelítés a fa előre-

```
AlfaBeta(p: pozicio; alfa,beta,melyseg: integer): integer;
var
    ertek,sz,t,m : integer;
    lepesek : array[1..MAX_SZELESSEG] of integer;
    lepesertek : array[1..MAX_SZELESSEG] of integer;

begin
    if melyseg = 0 then  { terminalis csomopont? }
        return( Pontertek(p) );
    sz := Lepesgenerator( lepesek );
    if sz = 0 then       { nincs legalis lepes? }
        return( Pontertek(p) );

    for i := 1 to sz do
        lepesertek[i] = TortenetiTabla [lepesek[i]];
    Rendez( lepesek, lepesertek);

    ertek := -oo;
    for m := 1 to sz do
        begin
            t := -AlfaBeta( p.lepes[m], -beta,-alfa,melyseg-1 );

            if t > ertek then
                ertek := t;

            if ertek >= beta then
                { Levagja a fa tobbi reszet }
                begin
                    legjobb := lepesek[i];
                    goto done;
                end;

            alfa := Max( alfa,ertek );
        end;

done:
    TortenetiTabla[legjobb] = TortenetiTabla[legjobb]+2*melyseg;
    return( ertek );
end.
```

1. táblázat

pozicio sorszama	terminalis csomopontok szama	csomopontok szama
1	1109	2994
2	24510	70606
3	7031	27431
4	10197	32928
5	49957	272771
6	6031	16459
7	18423	112066
8	4063	8850
9	20137	109633
10	23055	134317
11	23594	123901
12	7602	35946
13	37807	150213
14	8600	26519
15	8442	26307
16	6333	23319
17	9110	49189
18	29355	160101
19	10195	39295
20	34961	113035
21	35396	176939
22	17438	100742
23	11263	44100
24	11995	63969
osszesen	416604	1921630

2 fellepeses elemzes eseten

Ff8-d6	30	Ff8-e7	24
Ff8-b4	24	Ff8-h6	24
Ff8-a3	24	Hg8-h6	24
Ff8-g7	24	e6-e5	20
Ff8-c5	24	a7-a6	12
Hg8-e7	24	Bh8-h6	4

3 fellepeses elemzes eseten

Ff8-d6	72	Ff8-g7	24
Ff8-e7	32	Ff8-h6	24
Ff8-b4	24	Hg8-h6	24
Ff8-a3	24	e6-e5	20
Ff8-c5	24	a7-a6	12
Hg8-e7	24	Kc8-b8	8

4 fellepes elemzese eseten

Ff8-d6	156	Ff8-g7	24
Ff8-e7	32	Ff8-c5	24
Ff8-h6	28	Hg8-h6	24
Hg8-e7	26	a7-a6	22
Ff8-b4	24	e6-e5	20
Ff8-a3	24	Kc8-b8	8

5 fellepes elemzese eseten

Ff8-d6	480	Ff8-h6	56
a7-a6	390	Bd8-e8	54
Kc8-b8	332	Ff8-e7	54
Hg8-h6	146	Hg8-e7	38
Ff8-b4	68	Ff8-g7	26
e6-e5	64	Ff8-c5	24

6 fellepes elemzese eseten

a7-a6	1896	Ff8-b4	140
Ff8-d6	1580	Bd8-e8	130
Kc8-b8	1090	Hc6-b8	94
Hg8-h6	302	Ff8-e7	82
e6-e5	240	Hg8-e7	72
Ff8-h6	192	Ff8-g7	36

2. táblázat

rendezéséhez. Schaeffer a gyilkos heurisztikát a történeti heurisztika speciális eseteként emlegeti.

Schaeffer tanulmányában 24 különböző hadállás esetén mutatta be, hogy az általa kidolgozott heurisztikus eljárás milyen mértékben hatékony. Eredményeinek két legfontosabb adatát az *1. táblázatban* láthatjuk, mind a 24 pozicióra.

A *2. táblázat* azt mutatja, hogy az *ábrán* látható hadállás egyre mélyebb elemzése közben melyik lépés hányszor bizonyult a legjobbnak.

KOVÁCS P. ATTILA

ADOK — VESZEK — CSERÉLEK

Főben a rovatban rövid, szöveges, a mikroszámítógépekkel kapcsolatos hirdetéseket közlünk. A díjszabás: közleteknek gépelt soronként (60 karakter) 100,-Ft, magánszemélyeknek az első sor 50,-Ft, minden további sor 20,-Ft. Az NJSZT tagjainak az első három sor ingyenes. Hirdetéseiket a szerkesztőség címére várjuk.

ADOK

ATARI 600XL (16k) datasettel, 2 db joystickkel, játékprogramokkal eladó. Irányár: 10 000 Ft. Érdeklődni lehet: Vértes Balázs, Budapest, Leányka u. 3. 1221

ATARI 800XL garanciával eladó 22 000 Ft-ért. Levélcím: Ambrus Attila, Szeged, Korondi u. 4/A 6726. Tel.: 06/62/17-222

Enterprise 128k-ra táblakezelő program (egyszerű kezelés, bővíthető, grafikus lehetőségek) Varga Zoltán, Székesfehérvár, Tovölde u. 9/C. IV/2. 8000

C16 (komplett) 16 000 Ft-ért eladó. Tel.: 651-608

C64-es programokat adok-veszek-cserélek kazettán. Baráczsok Zsolt, Mezőkövesd, Szihalmi u. 1. II/1. 3400

Commodore 64 + VC 1541 meghajtó + **MPS 801** nyomtató eladó. Gép- és programdokumentáció, programokat tartalmazó lemezekhez kapcsolva (fordító, szövegszerkesztő, adatbáziskezelő, grafikai és zenei programok). Tel.: 790-144 az esti órákban

C64 + dataset + felhasználói és játékprogramok eladók. Reális ár! Tilés Elemér, vásárosnamény, Néphadsereg u. 5/a. 4800

Commodore VC20 rekámáron eladó, magnóval, joystickkal, tápegységgel, videomod., disassembler, grafikai bővítő, sok játék és egyéb programmal 9000 Ft-ért. Tóth E., Székesfehérvár, József Attila u. 2/C. 4/2. 8000

VC20 számítógép Commodore magnóval és dokumentációval eladó. Szabó László, Esztergom, Szentkirályi u. 56/5. 2500

ZX-Spectrum (48k) számítógépemet játékprogramokkal, dokumentációval olcsón eladom. Vitai János, Pócspetri, Felszabadulás u. 8. 4327

ZX-Spectrum (48k) magnóval, kb. 300 programmal eladó 20 000 Ft-ért. Jelinek András, Budapest, Mogyoródi u. 117. 1141

ZX81 (64k) programokkal és irodalommal olcsón eladó. Fábián Zoltán, Kecskemét, Szolnokihegy 34. 6000

Másoló és hibajavító program SFD 1001-es lemezes egységre. Teljes lemezmásolás, formálás, ellenőrzés, hibajavítás 6 percben. Bővebb felvilágosítás: Pivarnyik Attila vagy Szakál László, Miskolc-Egyetemváros, Vöröshadsereg u. 110. 3515, Tel.: 65-111

VESZEK

C64-et vagy C16-ot veszek magnóval (használtat is). Árajánlatokat a következő címre: Sebestyén Károly, Hajdúnánás, Mártírok u. 11. 4080

Primo B 64 számítógépet vennék, lehetőleg joystickkal. Vindics István, Mecseknádasd, Liszt Ferenc u. 39. 7695

CSERÉLEK

C16 programokat cserélek. Czikollai Erik, Vaskút, Szechenyi u. 19. 6521

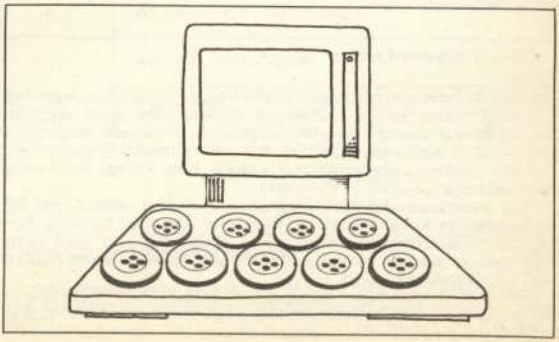
C64-hez német nyelvű GEOS leírást magyarra cserélnék. Magyar János, Szászhalombatta, Béke u. 8. III/2. 2440

C64-re felhasználói és játékprogramokat cserélek lemezen és kazettán is. Cserealapon kb. 800 program. A válaszokat listával kérem: Bicó András, Gorcsony, Hársfa u. 55. 7833

ZX-Spectrumra programok titkosítását, szivárványszínű és egyéb egyszerű betoltők megírását ingyen vállalom, cserébe játékprogramokat kérek. Szurmai Levente, Veszprém, Komarov u. 17. 8200

ZX-Spectrumra programokat cserélek. Válaszokat listával kérek: Vadon Zoltán, Budapest, Törcsvár u. 28. 1112

Ez a rovatunk **KODEX 2000** szövegszerkesztővel készült.



LASER és SELTRON

Fordítós játék

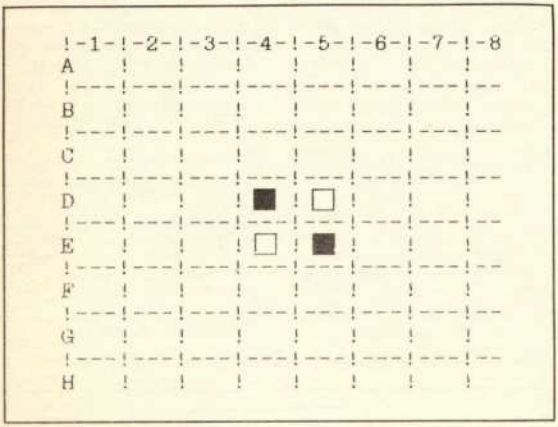
Papp Erikának és Papp Gábornak a Magazin 1987. júniusi számában közreadott levele és az olvasók időnként felbukkanó kérdései azt sejtetik, hogy az országban jónéhány Laser- és Seltron-tulajdonos van, akik — velem együtt — egyelőre nem „akarják" IBM PC-re cserélni szerény gépüket. Nekik küldöm az alábbi logikai játékot.

A fordítós játék (más néven Othello) lényege, hogy az *1. ábra* szerinti alaphelyzetből kiindulva a sötét és világos játékos úgy helyezze el figuráját a 8×8-as táblán, hogy ezáltal az ellenfél színét átló vagy vonal mentén közrefogja. A közrefogott szín ekkor a játékos saját színére vált. Az győz, aki a játék végén több négyzetet foglal el. Ha valaki nem tudja átfordítani ellenfele pozícióit, akkor kénytelen passzolni.

A program elindítása után a játékosok és a kezdő játékos nevét kell begépelni. Ezután a képernyőn az 1. ábra szerinti kiindulási helyzet látható. Az L betű leütésével megjelenik a soron következő játékos neve. Először a betű-, majd a számkoordinátát kell megadni. Helytelen karakter leütésekor vagy tiltott helyre lépés esetén hangjelzés figyelmeztet a hibára. Ha a soros játékosnak nincs olyan lépése, amivel ellenfelének legalább egy figuráját átfordítaná, akkor a P betűt üsse le, ezzel passzol. A V betűre a játék véget ér, és megjelenik az eredmény.

A játék tanulságos lehet a programozásban még nem nagyon jártas gyerekek számára, akik azért már tapasztalhatták, milyen könnyen áttekinthetetlenné válik egy hosszabb, BASIC nyelven írt program.

Ahhoz, hogy jól kezelhető, és ha nem is szigorúan programozáselméleti értelemben, de strukturált programot készíthessünk, érdemes a feladatot először nagy vonalakban, folyamatábra segítségével megtervezni (lásd a *2. ábrát*). A folyamatábra egyes négyzetei és rombuszai kisebb-nagyobb programokat rejtenek magukban. Nevezzük el ezeket moduloknak. A 2. ábra e modulok kapcsolatát írja le, s ránézve a programlistára, látható, hogy ezt egy rövid program valósítja meg (10—130-as sor).



1. ábra. A kiindulási helyzet

Egy programmodul általánosságban a *3. ábra* szerint épül fel. A bemenő és kimenő adatokat általános (globális) változók hordozzák, tehát ezeket más modulok is felhasználják. A helyi változók csak az aktuális modul működéséhez kellenek. Ilyen például egy ciklus indexváltozója.

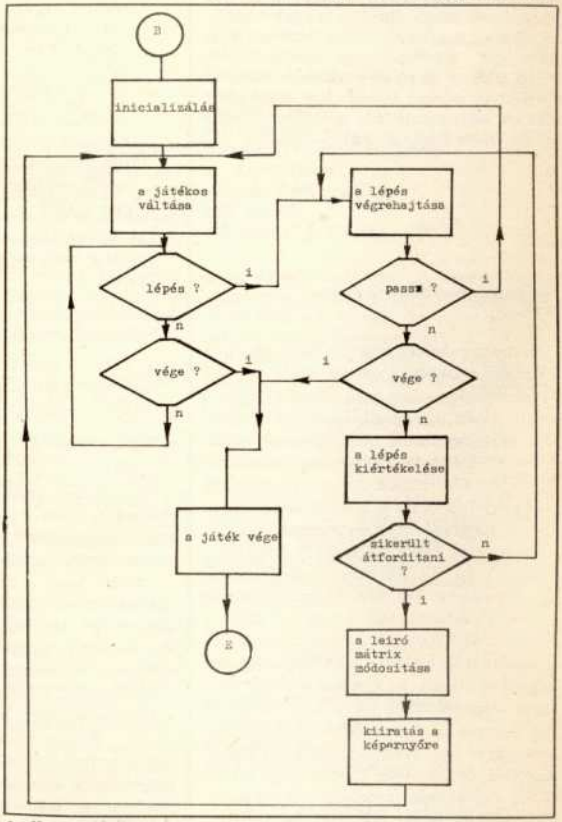
A BASIC-ben az egyes modulok úgy különíthetők el egymástól, hogy például 1000 többszöröseinek megfelelő sorszámon kezdődnek. A programfejlesztés szakaszában nem érdemes takarékoskodni.

```
1     '****************************************
2     '        FORDITOS JATEK
3     '****************************************
10    GOSUB9000          'INICIALIZALAS
20    FF=-FF
30    GOSUB3000          'VARAKOZAS KARAKTERRE
40    IFT$="V"THEN4000'A JATEK VEGE
50    IFT$<>"L"THEN30
60    GOSUB8000          'LEPES
70    IFT$="P"THEN20
75    IFT$="V"THEN4000
80    GOSUB7000          'KIERTEKELES
90    IFFF=0THEN60
100   GOSUB6000          'A LEIRO MATRIX MODOSITASA
110   GOSUB5000          'KIIRATAS A KEPERNYORE
120   GOTO20
130   END
135   '****************************************
3000  T$=INKEY$:T$=INKEY$:IFT$=""THEN3000
3010  RETURN
3115  '****************************************
4000  E1=0:E2=0
4010  FORI=0TO7:FORJ=0TO7
4020  IFA(I,J)=1THENE1=E1+1
4030  IFA(I,J)=-1THENE2=E2+1
4040  NEXT:NEXT
4050  IFE1=E2THEN4090
4060  IFE1>E2THENT1$=T1$ELSET1$=T2$
4070  PRINT@384,T1$;" GYOZOTT ";ABS(E1-E2);" PON
TTAL";
4080  GOTO4100
4090  PRINT@384,"DONTETLEN JATEK ";
4100  PRINT@448,"** UJ JATEK? (I/N) **"
4110  GOSUB3000
4120  IFT$="I"THENRUN10
4130  IFT$="N"THEN130
4140  GOTO4110
4145  '****************************************
5000  FORI=0TO7:FORJ=0TO7
5020  IFA(I,J)=1THENPRINT@(J*64+34+I*4),"■";
5030  IFA(I,J)=-1THENPRINT@(J*64+34+I*4),"□";
5040  NEXT:NEXT
5050  RETURN
5055  '****************************************
6000  IFE=0THEN6020
6010  FORI=1TOE:A(X0,Y0-I)=FF:NEXT
6020  IFK=0THEN6040
6030  FORI=1TOK:A(X0+I,Y0)=FF:NEXT
6040  IFD=0THEN6060
6050  FORI=1TOD:A(X0,Y0+I)=FF:NEXT
6060  IFNY=0THEN6080
6070  FORI=1TONY:A(X0-I,Y0)=FF:NEXT
6080  IFEK=0THEN6100
6090  FORI=1TOEK:A(X0+I,Y0-I)=FF:NEXT
6100  IFDK=0THEN6120
6110  FORI=1TODK:A(X0+I,Y0+I)=FF:NEXT
6120  IFDN=0THEN6140
6130  FORI=1TODN:A(X0-I,Y0+I)=FF:NEXT
6140  IFEN=0THEN6160
6150  FORI=1TOEN:A(X0-I,Y0-I)=FF:NEXT
6160  A(X0,Y0)=FF:RETURN
6165  '****************************************
7000  K=0:K=0:D=0:NY=0:EK=0:DK=0:DN=0:EN=0:FF=0
7010  IFY0<2THEN7100
7020  IFA(X0,Y0-1)=FFTHEN7100
7030  FORY=Y0-1TO0STEP-1
7040  IFA(X0,Y)=0THEN7100
7050  IFA(X0,Y)=FFTHEN7060ELSE7070
7060  E=Y0-Y-1:FF=1:GOTO7100
7070  NEXT
7100  IFX0>5THEN7200
7110  IFA(X0+1,Y0)=FFTHEN7200
7120  FORX=X0+1TO7
7130  IFA(X,Y0)=0THEN7200
7140  IFA(X,Y0)=FFTHEN7150ELSE7160
7150  K=X-X0-1:FF=1:GOTO7200
7160  NEXT
7200  IFY0>5THEN7300
```

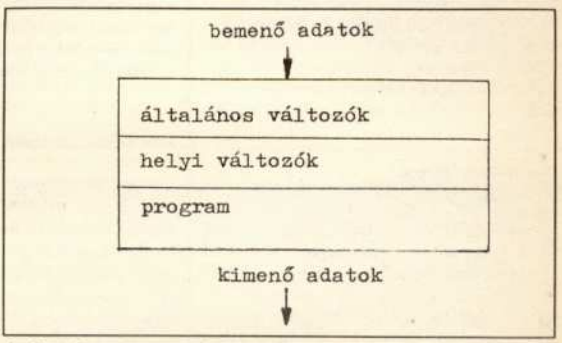
```
7210 IFA(XØ,YØ+1)=FFTHEN7300
7220 FORY=YØ+1TO7
7230 IFA(XØ,Y)=ØTHEN7300
7240 IFA(XØ,Y)=FFTHEN7250ELSE7260
7250 D=Y-YØ-1:FP=1:GOTO7300
7260 NEXT
7300 IFXØ<2THEN7400
7310 IFA(XØ-1,YØ)=FFTHEN7400
7320 FORX=XØ-1TOØSTEP-1
7330 IFA(X,YØ)=ØTHEN7400
7340 IFA(X,YØ)=FFTHEN7350ELSE7360
7350 NY=XØ-X-1:FP=1:GOTO7400
7360 NEXT
7400 IFYØ<2ORXØ>5THEN7500
7410 IFA(XØ+1,YØ-1)=FFTHEN7500
7420 IFYØ<7-XØTHENLI=YØELSELI=7-XØ
7430 FORI=1TOLI
7440 IFA(XØ+I,YØ-I)=ØTHEN7500
7450 IFA(XØ+I,YØ-I)=FFTHEN7460ELSE7470
7460 EK=I-1:FP=1:GOTO7500
7470 NEXT
7500 IFXØ>5ORYØ>5THEN7600
7510 IFA(XØ+1,YØ+1)=FFTHEN7600
7520 IF7-XØ<7-YØTHENLI=7-XØELSELI=7-YØ
7530 FORI=1TOLI
7540 IFA(XØ+I,YØ+I)=ØTHEN7600
7550 IFA(XØ+I,YØ+I)=FFTHEN7560ELSE7570
7560 DK=I-1:FP=1:GOTO7600
7570 NEXT
7600 IFXØ<2ORYØ>5THEN7700
7610 IFA(XØ-1,YØ+1)=FFTHEN7700
7620 IFXØ<7-YØTHENLI=XØELSELI=7-YØ
7630 FORI=1TOLI
7640 IFA(XØ-I,YØ+I)=ØTHEN7700
7650 IFA(XØ-I,YØ+I)=FFTHEN7660ELSE7670
7660 DN=I-1:FP=1:GOTO7700
7670 NEXT
7700 IFXØ<2ORYØ<2THEN7800
7710 IFA(XØ-1,YØ-1)=FFTHEN7800
7720 IFXØ<YØTHENLI=XØELSELI=YØ
7730 FORI=1TOLI
7740 IFA(XØ-I,YØ-I)=ØTHEN7800
7750 IFA(XØ-I,YØ-I)=FFTHEN7760ELSE7770
7760 EN=I-1:FP=1:GOTO7800
7770 NEXT
7800 IFFP=1THEN7820
7810 FORI=1TO3:SOUND31,1:NEXT
7820 RETURN
7825 '*****************************************
8000 IFFF=1THENT1$=T1$ELSET1$=T2$
8010 PRINT@384,"
  ":PRINT@384,T1$;
8015 '31 SPACE
8020 GOSUB3000
8030 IFT$="P"ORT$="V"THEN8085
8040 IFT$<"A"ORT$>"H"THEN8020
8050 PRINT" ";T$;:YØ=ASC(T$)-65
8060 GOSUB3000
8070 IFT$<"1"ORT$>"8"THEN8060
8080 XØ=ASC(T$)-49
8085 PRINTT$;
8090 FORI=1TO300:NEXT
8100 PRINT@384,"!---!---!---!---!---!---!---!---!---
-"
8105 IFT$="V"ORT$="P"THEN8120
8110 IFA(XØ,YØ)<>ØTHEN8130
8120 RETURN
8130 FORI=1TO3:SOUND31,1:NEXT
8140 GOTO8000
8145 '*****************************************
9000 CLS:COLOR2,Ø
9010 PRINT@168,"FORDITOS JATEK"
9020 PRINT@224,:INPUT"AZ ELSO JATEKOS NEVE (■)"
;T1$
9030 PRINT:INPUT"A MASODIK JATEKOS NEVE (□)";T2
$
9040 PRINT:INPUT"KI KEZD";T3$
9050 IFT3$=T1$THENFF=-1ELSEFF=1
9060 CLS
9070 A$="!-1-!-2-!-3-!-4-!-5-!-6-!-7-!-8"
9080 B$="!---!---!---!---!---!---!---!---!"
9090 C$="!   !   !   !   !   !   !   !   !"
9100 PRINTA$
9110 FORI=1TO7:PRINTC$:PRINTB$:NEXT
9120 PRINT C$;
9130 FORI=ØTO7:PRINT@(I*64+32),CHR$(I+1+64);:NE
XT
9140 DIMA(7,7):DIMTI$(16)
9150 A(3,3)=1:A(4,4)=1:A(4,3)=-1:A(3,4)=-1
9160 GOSUB5000
9170 RETURN
9175 '*****************************************
```

ni a REM-ekkel; fel kell írni a modulok nevét, a bemenő, helyi és kimenő változókat.

Egy jól szerkesztett program tehát modulokból áll és egy olyan részből, amely a modulok működését összerendezi. Az egyes modulokat legcélszerűbb szubrutinként szervezni. Ezt a módszert követve akár többen is nekiláthatnak egy feladat megoldásának.



2. ábra. A játék folyamatábrája



3. ábra. Egy programmodul felépítése

Egyikük megtervezi a fő folyamatábrát, a többiek pedig az egyes részfeladatok, a modulok programozásában munkálkodnak. Sok sikert!

PÁSZTOR FERENC

Az idén a nyaralásom egy kicsit hosszabbra sikerült, így az előző számok olvasói rovatának anyagát jó előre elkészítettem. Ennek azután az lett a következménye, hogy visszatérve a nyaralásból igen sok levél várt a szerkesztőségben. Ha csak részleteket közölnénk ezekből a levelekből, még akkor is meg tudnánk tölteni velük az egész lapot. Így elnézést kérek olvasóinktól, minden levélre itt nem tudunk válaszolni.

Nagy György, 924 01 Galanta

A saját, valamint több barátom nevében fordulok önökhöz. Csehszlovákiában 1986 decemberében jelent meg az üzletekben a SHARP MZ 821 típusú személyi számítógép. Mivel egyetemi tanulmányaink során nagyon jól tudjuk hasznosítani (számítástechnikát, kibernetikát tanulunk), ráadásul sokunknak hobbija is a számítástechnika, többen megvásároltuk ezt a gépet.

Szeretnénk kapcsolatba lépni e számítógép vagy az MZ 700 típusú számítógépek tulajdonosaival (az MZ 821-nek ugyanis kétféle üzemmódja van: az egyik a saját, a másik az MZ 700-ra írt programokat futtatja, illetve teszi lehetővé azok futását), ezért nagyon örülnénk, ha levelünket (vagy részletét) megjelentetnék lapjukban. Olyanokkal is szívesen felvennénk a kapcsolatot, akik ugyan nem rendelkeznek ilyen számítógéppel, de tudnának küldeni vagy eladni ehhez kapcsolódó irodalmat, cikkeket, információkat. Mivel már elég sok programunk van, lehetséges a programcsere és saját tapasztalataink átadása is. Segítségüket és az önök segítségét is előre nagyon köszönjük.

Reméljük, a felhívás nem marad visszhang nélkül.

Nemes Imre, Szeged

Nagy örömmel vettem kézbe a µM 1987/9. számát, s még nagyobb örömmel olvastam azon sorait, melyekben a LOGO alkalmazását javasolja. Mindenben maximálisan egyetértek. Sőt! Abban a szerencsés helyzetben vagyok, hogy merem állítani, hogy ha egy kicsike szakmai, (ismétlem, nem anyagi) segítséget kapnánk, akkor a µ'88-on már nemcsak elméleti, de *gyakorlati próbálkozásokat* is megbeszélhetnénk! Miért? Azért, mert —

— én 3 éve foglalkozom csak számítástechnikával, de elég intenzíven,
— most kezdtem el az egyetemet (JATE, Szeged) prog. mat. szakon, I/2 évfolyamos vagyok (L csoport),
— most akarok egy SZÁMÍTÁSTECHNIKAI TANFOLYAM-ot indítani a helyi művelődési központban,
— az iskolában és a művelődési központban együttesen 6 (!) VT TV-Computer van, sajnos lemezegység nincs, és csak egy magnó van. Tanfolyami terveim:

1. A TVC BASIC rajzoló utasításai
2. Írjunk egy olyan programot, amellyel könnyen lehet rajzolni! (LOGO FORDÍTÓ)
3. A LOGO teknőc-grafikáján át egyre bonyolultabb programozási példák felé!
4. A LOGO egyéb utasításai — szintén szükséglet szerint, tehát
5. FORTH, PASCAL, MPROLOG
6. ASSEMBLEREK — Z80, M65 XX
7. BASIC

Nem BASIC nyelvre kell megtanítani az embereket (hisz az egy nyelvtanfolyam), nem is programozni, hanem a *számítástechnika eredményeit felhasználni!*

Örülök, hogy a LOGO-val kapcsolatban egyetértünk, „drukkolok", hogy a Művelődési Központban szervezett tanfolyama sikerüljön. Egyetlen dologban nem egyezik a véleményünk, én nemcsak a „számítástechnika eredményeinek a felhasználását" tanítanám: a számítástechnika ennél sokkal „színesebb csokor". Én azt hiszem, hogy nekünk meg kell teremteni a lehetőséget ahhoz, hogy aki programozni akar tanulni (az sem baj, ha pont a mikroszámítógépek eszperantóját, a BASIC-et), az programozni tanulhasson, akit a hardver érdekel, az hardver szakértővé képezhesse magát, de aki „csak" szövegszerkesztésre akarja használni a számítógépet, az ehhez is megtalálja a lehetőséget.

Káli László, Jászapáti

Tavaly év vége óta foglalkozom számítástechnikával, és azóta olvasom az önök lapját. Ennek a rovatnak köszönhetem, hogy nagyszerű embereket ismerhettem meg. Eddig csak olvastam a kérdéseket, most viszont én szeretnék kérdezni, mert mástól nem remélhetek választ. Enterprise 128-as gépem van. A probléma a következő:

A Mikrovilágban az Enterprise-t bemutató cikkben olvastam, hogy 4690 forintos áron lehet a géphez kapni emulátort. A Centrum Áruházak közül viszont sehol sem tudnak arról, hogy az emulátor létezne. Tehát szeretném megtudni, hogy végül is van-e egyáltalán már ilyen, hogy kapható-e, és ha igen, akkor hol?

Levelének azért is örültem, mert szívügyem az „Olvasó írja" rovat, amelyről én is azt hiszem, hogy hasznos. A levelekből megismerhetjük egymás véleményét, és valóban nagyszerű és a számítástechnika ügye iránt elkötelezett emberekkel teremthetünk kapcsolatot. Ami az Enterprise-t illeti én is úgy tudom, hogy a meghirdetett Sinclair emulátor még nem kapható. A nyáron kb. 300 DM-ért láttam néhány Enterprise-t a müncheni Quelle kirakatában „árválkodni". Amikor az eladótól megkérdeztem, hogy lehet-e a géphez Sinclair emulátort kapni, úgy nézett rám, mintha egy Enterprise MS-DOS-t kértem volna.

Murcsó Károly, Tokod-altáró

Véletlenül hozzájutottam néhány C64-en futó fordítóprogramhoz, de leírás nélkül a kezelésük lehetetlen. Ezért kérem, írják meg, hogy a következő programokhoz hol tudok dokumentációt szerezni! (Ha nem tudják a választ, akkor kérem, hogy a levelem egy részét tegyék közzé!)

Tehát a programok: G-PASCAL, C64 FORTH, MAX-FORTH.

Közzétettük.

Göbölös László, Kalocsa

Régóta vagyok a Magazin olvasója. Nemrég kaptam a szüleimtől egy Enterprise számítógépet. Mivel eddig (2 és fél évig) Commodore gépeken dolgoztam, eleinte kicsit szokatlannak tűnt az Enterprise egyébként kiváló BASIC-je. A gép nagyon tetszik, meg vagyok vele elégedve, de a géphez ingyen járó, magyar nyelvű (!) Felhasználói kézikönyvnek gúnyolt könyvecskével már nem. A könyv több helyen hivatkozik az Enterprise technikai ismertetőre. Hát ilyet én nem kaptam. Létezik egyáltalán? Kérem, írja meg, hogy hol lehet kapni erre a gépre valamilyen értelmesebb könyvet! Utolsó kérésem az lenne, hogy ha tudnának Enterprise-tulajdonosról, írják meg a címét!

Az Enterprise-tulajdonosoknak szóló javaslatom: Enterprise-tulajdonosok, egyesüljetek! A szándék megvalósítására ajánlom, keressék meg az NJSZT HCC-t.

KOVÁCS GYŐZŐ

Steigers, J.:
A robotok és a Commodore 64
(Budapest, 1987.
Data Becker–Novotrade,
128 oldal. Ára: 249,— Ft.)

Napjainkban egyre szélesebb körben terjednek a különböző feladatok ellátására kifejlesztett automaták és robotok. Alkalmazásuk célja elsősorban az emberi munka megkönnyítése, vagy egy-egy részének gazdaságos kiváltása. Bár a könyv gyakorlatias szellemben készült, mégis nyújt némi betekintést a kibernetika, a vezérlés- és szabályozástechnika területére. Röviden összefoglalja az embert utánzó szerkezetek, az automaták történetét is.

A kötet tartalmazza a mechanika, az elektronika és a programozástechnika alapszintű ismereteit, amelyek elegendőek ahhoz, hogy a C64 számítógépből házi robotot építhessen az olvasó. A leírtak megértése az elektronika és a forrasztás alapszabályain kívül más előismeretet nem igényel. A bemutatott kapcsolásokat általában könnyű elkészíteni, és megépítésükhöz nem szükséges magának a C64-es alapgépnek az átalakítása. A könyvben szó esik arról is, hogyan lehet egy robotot a C64-gyel összekötő kábel nélkül programozottan vezérelni.

Blume, Ch.—Jakob, W.:
Ipari robotok
programozási nyelvei
(Budapest, 1987.
Műszaki Könyvkiadó,
274 oldal. Ára: 127,— Ft.)

Az ipari robotokról eddig megjelent könyvek a programozás, illetve a programozási nyelvek kérdéseit csupán érintőlegesen tárgyalták. Hiányzott az egyes programnyelvek teljesítőképességének és szolgáltatási jellemzőinek összehasonlítása is. A kötet az erre a célra kialakított rendszerező elvek segítségével összehasonlítja az ipari robotok vezérlésére alkalmazott nyelveket. Öt programozási nyelvet elemez: az AL, a VAL, a HELP, a SIGLA és a ROBEX nyelveket. Nem tekinthető a tárgyalt nyelvek elsajátítására alkalmas tankönyvnek, sokkal inkább az ipari robotok programozásának specifikus problémái és eljárásai iránt kíván érdeklődést ébreszteni.

Encarnacao, J.—Schlechtendahl, E. G.:
CAD, számítógéppel
segített tervezés
(Budapest, 1987.
Műszaki Könyvkiadó,
339 oldal. Ára: 170,— Ft.)

Az utóbbi évtizedekben lejátszódó világgazdasági folyamatok, a felgyorsult műszaki-technikai haladás következtében jelentős mértékben megváltoztak a termékekkel szemben támasztott követelmények, és világszerte egyre jobban érezhető a törekvés a nagyobb hozzáadott értéket képviselő dolgok előállítására. Előtérbe kerültek a gazdaságos anyag- és energiafelhasználásra, az újszerűségre és a piaci igényekre való gyors reagálás követelményei.

Mindezekkel párhuzamosan megváltozott a mérnöki munka feltétel- és eszközrendszere is. A számítógépek megjelenése és elterjedése óta a számítógépes „technológia", ezen belül a számítógéppel segített tervezés (CAD) is sokat fejlődött és elméleti megalapozottságú interdiszciplináris tudományággá vált, amelynek gyakorlati használhatóságát a CAD rendszerek ipari méretű bevezetése és alkalmazása bizonyítja.

A tervezési tevékenység ma már megkívánja és a technika fejlettsége lehetővé teszi olyan számítógéppel segített tervező rendszerek kialakítását, amelyek integrálják az adat- és információtárolási, a mérnöki elemzési, a grafikus modellezési és megjelenítési funkciókat, azonkívül szervesen illeszkednek a tervezési környezetben a termelési folyamatba. Az ilyen rendszerek fejlesztésének, üzemeltetésének és hatékony alkalmazásának a gyakorlati elsajátításon kívül egyik elengedhetetlen feltétele a CAD elméletének, szemléletmódjának és eszköztárának megismerése.

Ennek a követelménynek tesz eleget ez a könyv, amely igen átfogóan, korszerű szemlélettel, a tervezéselméletre és a számítástechnika tudományára támaszkodva — már a megfelelő alapismereteket és jártasságot feltételezve — tárgyalja a számítógéppel segített tervezés elméleti és gyakorlati kérdéseit. A kötetet a közölt részletes irodalomjegyzék és a gyakorlati életből vett példaanyag teszi teljessé. A számítógéppel segített tervezéssel ilyen összefüggésben foglalkozó szakkönyv hazánkban eddig még nem jelent meg.

Pajor Gábor:
Az IBM PC-ről
kezdő felhasználóknak
2. A szoftver
(Budapest, 1987.
LSI ATSZ,
68 oldal. Ára: 70,— Ft.)

Ez a könyv a sorozat „A hardver" című kiadványát követi. Az IBM PC-ket a szoftveren keresztül mutatja be, mivel így gyorsabban és érdekesebben tárul ki a kép a gép alkalmazási területeiről és lehetőségeiről. Bár a szerző nem tűzte ki célul egy-egy program vagy programcsomag részletes ismertetését, az egyes programokat addig a mélységig mutatja be, amíg azoknak a gép lehetőségeinek megismertetésében szerepük van.

Ez a kötet is segíteni kívánja az igényes vásárlókat, hogy vásárlásuk tárgyát még beszerzés előtt megismerhessék; vagyis azoknak szól, akik előre tisztázni akarják, hogy az adott feladat elvégzésére a gép alkalmas lesz-e.

Lángos István:
Bevezetés
az IBM PC XT/AT DOS-ba
(Budapest, 1987.
Novotrade,
127 oldal. Ára: 99,— Ft.)

A kötet tanfolyamok alapján született. Nem akarja pótolni a DOS-leírást és az egyéb IBM-dokumentációkat, nem törekszik teljességre. Csupán biztatásul kínálkozik azoknak, akik IBM PC XT/AT vagy más IBM kompatibilis gép mellé először ülnek le. Az olvasókról a szerző valamilyen számítógépes előéletet már feltételez.

Minden hétfőn 17-től 19 óráig

a Mikroszámítógép Magazin munkatársai és felkért szakértők válaszolnak az olvasók kérdéseire a szerkesztőségben: Budapest II., Fő u. 68. I. em. 109. vagy a 154-090 és a 154-250-es telefonon.

Minden kedden 17-től 20 óráig
ENTERPRISE-klub
a VSZM
Közösségi Házban
(Bp. XI., Fehérvári út 120.)
Klubvezető: Romvári Gábor
Telefon: 450-950/473

A bakui Korvet

Új iskolaszámítógép-típus jelent meg a Szovjetunióban: a bakui Rádióépítési Termelői Egyesülés bemutatta a Korvet nevű mikroszámítógépét. Az Intel 8080 szovjet funkcionális megfelelőjén alapuló gép operációs rendszere a CP/M 2.2 verziónak megfelelő mikroDOS. Ez gazdag programnyelvi — Pascal, FORTH, Logo, Prolog — és alkalmazóiszoftver-választékot biztosít az új típusnak. A gép kiépítettsége kategóriájához képest viszonylag nagy, hiszen a 64 kbájt operatív tárat hajlékonylemezes tároló egészíti ki, és a nyomtatócsatlakoztatás lehetősége is megvan. A gépet kétféle változatban gyártják: a PK8020 jelzésű a tanári gép, a PK8010-es pedig a diákoké. Ebből már következik is a gépek hálózatba fűzhetősége: maximum 15 diákgépet köthetnek össze egy tanári géppel. Az utóbbiról folyamatosan követhető és irányítható a diákok munkája.

A teljes egészében szovjet alkatrészekből álló gépből az idén százezres sorozat előállítását tervezik.

Korvet gépekkel felszerelt tanterem

Szoftverlopás bolgár módra

Bulgáriában is lezajlott az első szoftverlopási per. A Szisztemizot kifejlesztette a DIAMSZ nevű hálózati szoftvert a PDP-kompatibilis minigépekhez, amelynek igen nagy volt a piaci sikere. Egyszer aztán jelentkezett az Energetikai Intézet: „Megvásároltuk a DIAMSZ rendszert az Újítási Intézettől. Az installálta is, csak nem teljesen a felhasználói elképzelések szerint. Nem tudnának-e a Szisztemizot szakemberei segíteni?" A DIAMSZ szerzőinek aztán már a helyszínen csupán percek kérdése volt annak megállapítása, hogy az általuk fejlesztett szoftvert az Újítási Intézet továbbértékesítette. Az ügy a bíróságon folytatódott, amely magas kártérítést ítélt meg a Szisztemizot számára. Az eset kapcsán a Szisztemizot szakemberei hangsúlyozták, hogy végtelenül egyszerű az embernek saját munkáját felismerni.

Elembázis

Bár a világpiacon 1987-ben váltották fel a 256 kbites operatív tármorzsákat az 1 Mbitesek, a szocialista országok ettől egyelőre még messze vannak: a csúcsot eddig a 16 kbites tármorzsák jelentették. Az elmúlt ősz azonban több tekintetben is újdonságokat hozott. A szerényebbek közé tartoznak az NDK új termékei, a 32 kbites statikus RAM és a szintén 32 kbites EPROM. Szakmai körökben azonban inkább a két szovjet újdonság keltett feltűnést. Bemutatták ugyanis a 64 kbites, 64 × 1 bit szervezésű, KM132RU10A típusjelzésű statikus RAM-ot és a 256 kbites, K565RU7 típusú dinamikus RAM-ot. Ez utóbbi a Szovjetunióban komoly lendületet adhat a nagy operatív tárú professzionális mikroszámítógépek gyártásának.

Környezet-értékelés

Az MTA Földrajztudományi Kutatóintézetének munkatársai olyan mikroszámítógépes környezetértékelési rendszert fejlesztettek ki, amellyel kimutatják, hogy az egyes növények termesztésére melyik tájegység a legalkalmasabb. A kutatók elsőként Somogy megye termőterületeit vizsgálták, majd a tavalyi év során Borsod és Szolnok megye került sorra.

Összegyűjtötték a területek éghajlati, domborzati, geológiai, talajtani és hidrológiai adottságaira vonatkozó adatokat: 54 legfontosaub környezeti tényezőt figyelembe véve, 16 négyzetkilométerenként minősítették a termőföldeket. A szolgáltatások közé tartozik olyan térképek készítése, amelyek az egyes növények és gyümölcsök termeléséhez a rendelkezésre álló területeket tíz szín alkalmazásával rangsorolják. Somogy megyében például 19 növény-, illetve gyümölcsfajta termőhelyeit szemléltetik.

A földrajzi környezeti információs rendszer azonban nemcsak a megyék átfogó minősítésére alkalmas. Ennél jóval részletesebb, kisebb földterületekre, akár táblákra lebontható térképek is készíthetők, amelyek jelentősen segíthetik a nagyüzemek vezetőit a döntéseik meghozatalában.

A mikroszámítógépes rendszert alkalmassá tették a környezetgazdálkodásra vonatkozó más adatok rendszerezésére is: például segítségével jól feltérképezhető, hogy hazánk mely részeit sújtja leginkább a környezetszennyezés.

Vasi terv

Széles körű összefogással javítják Vas megye középiskoláinak számítógépekkel való ellátását. Az iskolaszámítógép-program keretében a gépek árának harmadát a Tudományszervezési és Informatikai Intézet, további harmadát a Vas Megyei Tanács adja. Az iskolának tehát mindössze az ár harmadik harmadát kell fizetnie, azaz egy Commodore Plus/4 gépnél 3700 forintot. Az akció a jelenlegi ötéves tervidőszakra szól, és első lépcsőként szeptemberben már 250 számítógépet adtak át a megye középiskoláinak. A tervek szerint a következő ötéves tervben is folytatják az akciót, és akkor majd az általános iskolákat látják el számítógépekkel.

Liftügyelet

Központi liftügyeletet szervezett a Szegedi Ingatlankezelő Vállalat. A város 142 felvonójának üzemeltetését óránként ellenőrzi a C64-es gépre épülő rendszer. A lifteket légvezetéken 34 alközpontba kapcsolják, amelyekből rádión továbbítják a jeleket az azokat fogadó számítógépbe. Az esetleges liftben rekedéseket is ez a jelző rendszer észleli, és éjjel-nappal két gépkocsi áll készenlétben a kimentésre. A liftek rossz üzembiztonságát jelzi, hogy éjszakánként általában 2-3, és nappal is átlag három vészjelzés érkezik.

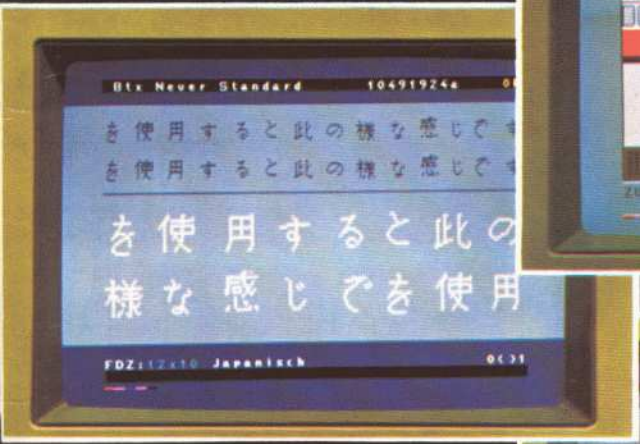
Az 1985 szeptembere óta eredményesen működő rendszer kiépítése tízmillió forintba került. A megtérülést gyorsítandó, a szövetkezeti házak liftjeinek 24 órás ügyeletét is vállalják, havi 1750 forintért.

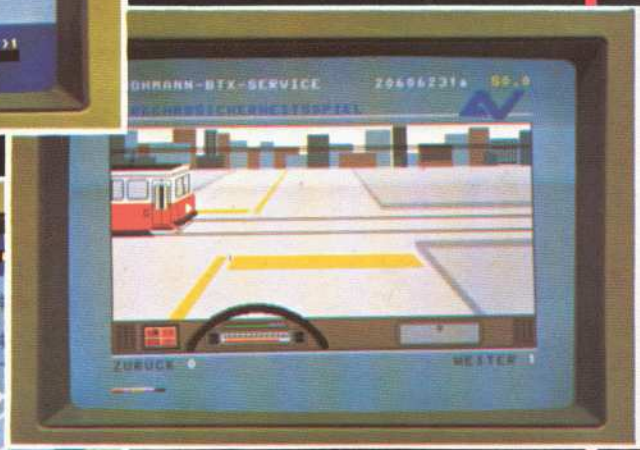
Mi van a csomag körül?

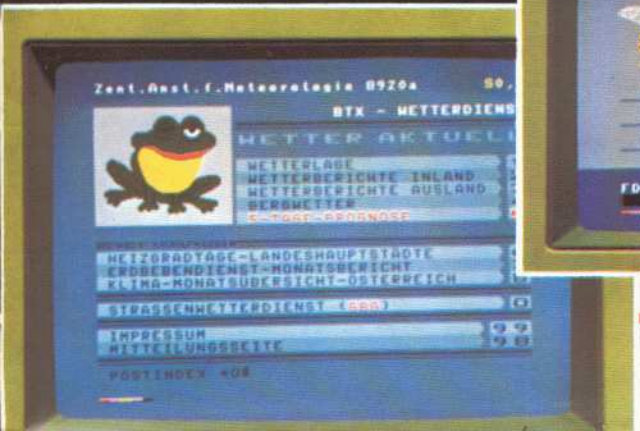
A General Electricnél számítógépes programot készítettek a csomagoláshoz szükséges habanyag formaidomok igénybevételtől függő tervezéséhez és képernyős megjelenítéséhez. A program figyelembe veszi az összes lehetséges igénybevételt, minden goromba behatást, majd ennek alapján határozza meg az alkalmas csomagolóanyagot és a szükséges méreteket. A csomagolás az eddigihez képest fele idő alatt megtervezhető.

Btx Never Standard
Albert Einstein

Btx Never Standard
DER LANGE WEG DES WASSERS ZUM MEER


Btx Never Standard
を使用すると此の様な感じで
を使用すると此の様な感じで
を使用すると此の
様な感じでを使用
FDZ:12x10 Japanisch


ZURUCK 0
WEITER 1


Zent.Anst.f.Meteorologie 8920a
BTX - WETTERDIENS
WETTER AKTUELL
WETTERLAGE
WETTERBERICHTE INLAND
WETTERBERICHTE AUSLAND
BERGWETTER
HEIZGRADTAGE-LANDESHAUPTSTADTE
ERDBEBENDIENST-MONATSBERICHTE
KLIMA-MONATSUBERSICHT-OSTERREICH
IMPRESSUM
MITTEILUNGSSEITE

normale Gr.
doppelte Gr.

Az 1987. évi tavaszi BNV-n díjat kapott

KODEX 2000

szövegszerkesztő berendezés kurzororientált, dinamikus képernyőformázó és menüvezérelt rendszer.

Kezelése egyszerű, kényelmes és nem igényel semmiféle számítástechnikai képzettséget.

Szoftvere folyamatosan bővül.

Fejlesztette: **BME Villamosmérnöki Kar, Folyamatszabályozási Tanszék,**
Budapest, Műegyetem rkp. 3/9. 1111.
Tel.: 453-500

Gyártja: **KONTAKTA Alkatrészgyár Ózdi Gyáregysége,**
Ózd, Bolyki főút 82. 3600. Tel.: 47/12-555

Forgalmazza: **MIGÉRT Író- és Számológéposztály,**
Dimitrov tér 14. 1093. Tel.: 175-081

Szervizét ellátja: **ITV KODEX Szakszervize,**
Budapest, Bécsi u. 8/10. 1052. Tel.: 329-531
Budapest, Báthori u. 5. 1054. Tel.: 121-090